

【特許請求の範囲】

【請求項1】 主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませるデータファイルを提供する電子情報付与装置において、

前記主要情報を構成するデータ群の中の一部のデータを前記付属情報を構成するデータに基づいて変更するために、前記主要情報における変更すべき前記一部のデータに対して、所定のアルゴリズムに従う演算を、前記付属情報を構成するデータと外部から設定可能な補助情報とを用いて行なう手段を具備する電子情報付与装置。

【請求項2】 主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませるデータファイルから前記付属情報を抽出する若しくは前記主要情報を修復する電子情報検出装置において、

前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な鍵情報の少なくとも一部を、外部から供給される情報として受け取る鍵情報受取手段を具備し、受け取った鍵情報を利用して前記付属情報の抽出若しくは前記主要情報の修復を行なうようにしたことを特徴とする電子情報検出装置。

【請求項3】 主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませるデータファイルを提供する電子情報付与方法において、

前記主要情報を構成するデータ群の中の一部のデータを前記付属情報を構成するデータに基づいて変更するために、前記主要情報における変更すべき前記一部のデータに対して、所定のアルゴリズムに従う演算を、前記付属情報を構成するデータと外部から設定可能な補助情報とを用いて行なうステップを具備する電子情報付与方法。

【請求項4】 主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませるデータファイルから前記付属情報を抽出する若しくは前記主要情報を修復する電子情報検出方法において、

前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な鍵情報の少なくとも一部を、外部から供給される情報として受け取るステップを具備し、受け取った鍵情報を利用して前記付属情報の抽出若しくは前記主要情報の修復を行なうようにしたことを特徴とする電子情報検出方法。

【発明の詳細な説明】**【0001】**

【発明の属する技術分野】この発明は電子的記憶媒体に

楽音制御情報(MIDIデータなど)などの主要情報と付属情報とを暗号化処理して記憶したり、記憶媒体に記憶されている暗号化処理された主要情報と付属情報に基づいて元の主要情報及び付属情報を検出する電子情報付与及び検出のための装置及び方法に関する。

【0002】

【従来の技術】最近では、パーソナルコンピュータを使用して、ユーザ自身が音楽データ、映像データ及び波形データなどを作成したり、それらに種々の変更を加えたりすることが容易にできるようになった。従って、パーソナルコンピュータを使用することによって、市販のFD、CD-ROM、LDなどの記憶媒体に記録された音楽データ、映像データ及び波形データなどを自由に読み出して、それらに種々の変更処理を加えたりすることができる。市販のCD-ROMやLDなどに記録されているデータは、その販売者や製作者などに著作権があるため、本来それらを自由に改変することは、著作権侵害の観点から許されるものではない。従って、現在では、CD-ROMやLDなどの媒体に記録されている音楽データ、映像データ又は波形データなど主データの著作物がだれに属するものなのかを示すための著作権表示データを、その音楽データ、映像データ又は波形データなど主データを記録する主データ記録部とは別のヘッダ部において付属情報として付加的に記録することによって、著作権者を明示し、著作権侵害の未然防止を図っているのが現状である。また、付属情報には、このような著作権表示データの他にも、その音楽データ、映像データ及び波形データの題名などを示す情報、又はその映像データや波形データなどがどのようなデータ圧縮技術で圧縮されているのかなどの記録形式を示す情報などがある。

【0003】

【発明が解決しようとする課題】しかしながら、現在では、パーソナルコンピュータなどを使って自由にデータを書き換えたり変更したりすることができるので、故意又は過失によって、著作権表示データや種々の付属情報が削除されたり、書き換えられたりするという問題がある。特に、これらの付属情報がヘッダ部に記録されている場合は、削除や改竄が容易になされてしまうという問題がある。また、最近ではネットワーク通信の発達に伴って、このように著作権表示データの削除や改竄がなされた音楽データ、映像データ又は波形データなどが通信ネットワークを介して広く流通するという問題も起こり易い。この発明は、音楽データ、映像データ又は波形データなどのような主要情報のデータフォーマットを変更することなく付属情報を付加すると共にこれらのデータに暗号化処理を施し、この暗号を解読しない限り、これらの主要情報や付属情報を再生して利用することのできないようにした電子情報付与及び検出装置及び方法に関するものである。

【0004】

【課題を解決するための手段】この発明に係る電子情報付与装置は、主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませるデータファイルを提供する電子情報付与装置において、前記主要情報を構成するデータ群の中の一部のデータを前記付属情報を構成するデータに基づいて変更するために、前記主要情報における変更すべき前記一部のデータに対して、所定のアルゴリズムに従う演算を、前記付属情報を構成するデータと外部から設定可能な補助情報とを用いて行なう手段を具備するものである。前記補助情報は外部から任意に設定可能であり、設定された該補助情報は、前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な「鍵情報」となりうものである。これにより、解読の際の鍵情報となる前記補助情報そのものは、データファイルのデータ内容とは無関係に任意に設定されるから、該データファイルの中には直接的にはそれと判るように含まれていないものとなる。なお、付属情報を主要情報の中に潜在的に組み込むためのアルゴリズム演算に際して使用する補助情報としては、データを暗号化するための演算に使用する変数情報や、データを暗号化するための演算を行う際の種々の条件を設定する変数情報若しくは条件設定情報など、その他、付属情報を主要情報の中に潜在的に組み込むためのアルゴリズム演算に関連して使用される任意の情報であってよい。

【0005】この電子情報付与装置は、主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませるデータファイルを提供するものであり、換言すれば、主要情報の中に付属情報を潜在的に埋め込み、主要情報に対して結果的に暗号化処理を施すことになるものである。一例として、主要情報としては、MIDIデータのキーオンイベントデータ、プログラムチェンジデータ又はコントロールチェンジデータ、波形データ、又は映像データなどがある。付属情報としては、著作権名、曲の題名、画像／映像の題名などに関する文字データや波形データの圧縮方法などのデータ形式に関するデータやその他の種々のデータ（暗号文、鍵情報、ID、パスワード、ニュース文）などがある。この発明に係る電子情報付与装置（換言すれば「エンコーダ」）と電子情報検出装置（換言すれば「デコーダ」）とを総称して電子情報処理システムと呼ぶとすると、この電子情報処理システムでは、一例として、主要情報を構成するデータ群の中の所定のデータ群として、MIDIデータの場合にはキーオンイベントデータ群を用い、このキーオンイベントデータ群の中の単位データとして、ベロシティデータやキーコードデータを用い、これらのデータのうち所定のデータを付属情報のデータ部分に応じて分散的に変更し、主要情報のデータ群の中

に付属情報のデータを分散的に潜ませる（埋め込む）ことで、付属情報を主要情報のデータ群の中に潜ませてなるデータファイルを提供する。

【0006】このデータファイルから前記付属情報を抽出する若しくは前記主要情報を修復するためには、解読のために所定の「鍵」情報が必要となるが、そのための「鍵」情報それ自体が該データファイルの中に含まれているような記録形態もありうる。しかし、それでは、オーソライズされていない者が、不当に当該データファイルの再生を行ないまたその付属情報を解読しようとする場合に、当該データファイルの中から解読のために必要な所定の「鍵」情報を容易に取得してしまうおそれがあり、セキュリティ面での信頼性に欠けるものとなってしまふ。そこで、この発明では、所定のアルゴリズムに従う演算を前記付属情報を構成するデータと外部から設定可能な補助情報とを用いて行なうようにしたことにより、前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な「鍵情報」の一種となる前記補助情報そのものは、データファイルのデータ内容とは無関係に任意に設定されることで、該データファイルの中には直接的にはそれと判るように含まれていないものとなるようにしたことを特徴としたものである。これにより、オーソライズされていない者が、不当に当該データファイルから主要情報の修復を行ないまたその付属情報を解読しようとしたとしても、肝心のデータ修復用「鍵情報」をデータファイルからは得ることができないことにより、データ修復若しくは解読を行なうことができないことになり、暗号化又は電子透し埋め込みによる機密的な情報記録に際して、セキュリティ機能を向上させることができる、という優れた効果を奏する。

【0007】この発明に係る電子情報検出装置は、主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませるデータファイルから前記付属情報を抽出する若しくは前記主要情報を修復する電子情報検出装置において、前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な鍵情報の少なくとも一部を、外部から供給される情報として受け取る鍵情報受取手段を具備し、受け取った鍵情報を利用して前記付属情報の抽出若しくは前記主要情報の修復を行なうようにしたことを特徴とするものである。

【0008】このように、前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復する機能を有する電子情報検出装置の側において、鍵情報受取手段を具備することにより、前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な鍵情報の少なくとも一部を、外部から供給される情報として受け取り、この受け取った鍵情報を利

用して前記付属情報の抽出若しくは前記主要情報の修復を行なうことができるようになる。すなわち、上述と同様に、データファイルの中には必要な鍵情報が含まれていないことになり、オーソライズされていない者が、不当に当該データファイルから主要情報の修復を行ないまたその付属情報を解読しようとしたとしても、肝心のデータ修復用「鍵情報」をデータファイルからは得ることができないことにより、データ修復若しくは解読を行なうことができないことになり、暗号化又は電子透し埋め込みによる機密的な情報記録に際して、セキュリティ機能を向上させることができる、という優れた効果を奏する。鍵情報受取手段に対する鍵情報の与え方は、どのようなやり方を用いてもよい。例えば、データファイルとは別途にデータ通信等により電子情報の形で自動的に与えるようにしてもよいし、オーソライズされた利用者に対してのみ印刷物または電話等で通知することに基づき該利用者が該鍵情報のデータ入力操作を手動で行なうようにしてもよいし、様々なやり方であってよい。

【0009】なお、付属情報を主要情報の中に潜在的に組み込むためのアルゴリズム演算に際して使用する「補助情報」、若しくは前記付属情報を抽出する又は前記主要情報を修復するのに必要とされる「鍵情報」は、一種類とは限らず、複数種類あってよい。例えば、電子情報付与装置（エンコーダ）において、前記主要情報を構成するデータ群の中の一部のデータを前記付属情報を構成するデータに基づいて変更するために、前記主要情報における変更すべき前記一部のデータに対して、所定のアルゴリズムに従う演算を、前記付属情報を構成するデータと所定の変数とを用いて行なうようにした場合、その解読（デコード）のためにアルゴリズムのみならず、該所定の変数も必要であり、この所定の変数に対応する情報が上記補助情報若しくは鍵情報の一種として使用されることとなる。あるいは、前記主要情報を構成するデータ群の中の一部のデータを前記付属情報を構成するデータに基づいて変更するためのデータ編集処理を行なうときに、特定の編集条件を設定して該データ編集処理を行なったような場合も、その解読（デコード）のためには設定された編集条件が何であったかが判っている必要があり、この編集条件を設定する情報も上記補助情報又は鍵情報の一種として使用されることとなる。

【0010】この発明は、装置発明として構成し、実施することができるのみならず、方法発明として構成し、実施することもできる。また、この発明は、コンピュータプログラムまたはDSP等のためのマイクロプログラムの形態で実施することができるし、そのようなコンピュータプログラムまたはマイクロプログラムを記憶した記録媒体の形態で実施することもできる。

【0011】

【発明の実施の形態】以下、この発明の実施の形態を添付図面に従って詳細に説明する。図2は、この発明に係

る電子情報処理システムとして動作する電子楽器の全体構成を示すブロック図である。マイクロプロセッサユニット（CPU）21はこの電子楽器全体の動作を制御するものである。このCPU21に対して、バス2Mを介して各種のデバイスが接続される。

【0012】CPU21はROM22及びRAM23内の各種プログラムや各種データ、及び外部記憶装置から取り込まれた楽音制御情報（MIDIデータ）に基づいて全体の動作を制御すると共に取り込まれたMIDIデータ内に付属情報を分散して記憶したり、付属情報の分散記憶されたMIDIデータをフロッピーディスクから取り込み、それから付属情報を検出したりする。この実施の形態では、外部記憶装置として、フロッピーディスクドライブ24、ハードディスクドライブ25、CD-ROMドライブ26などを例に説明するが、これ以外の光磁気ディスク（MO）ドライブ、PDDライブなどを用いてもよい。また、通信インターフェイス27を介して通信ネットワーク28上のサーバコンピュータ29などから楽音制御情報などの各種情報などを取り込んでもよいし、MIDIインターフェイス2Aを介して他のMIDI機器2BなどからMIDIデータなどを取り込んでもよい。CPU21は、このような外部記憶装置から取り込まれたMIDIデータや鍵盤2Cの押鍵操作に基づいて生成したMIDIデータを音源回路2Jに供給する。なお、外部に接続された音源回路を用いて発音処理を行うようにしてもよい。

【0013】ROM22はCPU21の各種プログラム（システムプログラムや動作プログラムなど）や各種データを格納するものであり、リードオンリーメモリ（ROM）で構成されている。RAM23は、CPU21がプログラムを実行する際に発生する各種データを一時的に記憶するものであり、ランダムアクセスメモリ（RAM）の所定のアドレス領域がそれぞれ割り当てられ、レジスタ、バッファ、フラグ、テーブル等として利用される。

【0014】また、ハードディスク装置25などの外部記憶装置に前記動作プログラムを記憶するようにしてもよい。また、前記ROM22に動作プログラムを記憶せずに、ハードディスク装置25などの外部記憶装置にこれらの動作プログラムを記憶しておき、それをRAM23に読み込むことにより、ROM22に動作プログラムを記憶したときと同様の動作をCPU21に行わせるようにしてもよい。このようにすると、動作プログラムの追加やバージョンアップ等が容易に行える。着脱自在な外部記憶媒体の1つとして、CD-ROMを使用してもよい。このCD-ROMには、自動演奏データやコード進行データや楽音波形データや映像データなどの各種データ及び任意の動作プログラムを記憶していてもよい。CD-ROMに記憶されている動作プログラムや各種データは、CD-ROMドライブ26によって、読み出さ

れ、ハードディスク装置25に転送記憶させることができる。これにより、動作プログラムの新規のインストールやバージョンアップを容易に行うことができる。

【0015】なお、通信インターフェイス27をデータ及びアドレスバス2Mに接続し、この通信インターフェイス27を介してLAN（ローカルエリアネットワーク）やインターネット、電話回線などの種々の通信ネットワーク28上に接続可能とし、他のサーバコンピュータ29との間でデータのやりとりを行うようにしてもよい。これにより、ハードディスク装置25内に動作プログラムや各種データが記憶されていないような場合には、サーバコンピュータ29からその動作プログラムや各種データをダウンロードすることができる。この場合、クライアントとなる楽音生成装置である自動演奏装置から、通信インターフェイス27及び通信ネットワーク28を介してサーバコンピュータ29に動作プログラムや各種データのダウンロードを要求するコマンドを送信する。サーバコンピュータ29は、このコマンドに応じて、所定の動作プログラムやデータを、通信ネットワーク28を介して自動演奏装置に送信する。自動演奏装置では、通信インターフェイス27を介してこれらの動作プログラムやデータを受信して、ハードディスク装置25にこれらのプログラムやデータを蓄積する。これによって、動作プログラム及び各種データのダウンロードが完了する。

【0016】なお、本発明は、本発明に対応する動作プログラムや各種データをインストールした市販のパーソナルコンピュータ等によって、実施させるようにしてもよい。その場合には、本発明に対応する動作プログラムや各種データを、CD-ROM26やフロッピーディスク等の、パーソナルコンピュータが読み込むことができる記憶媒体に記憶させた状態で、ユーザーに提供してもよい。そのパーソナルコンピュータが読み込むことができる記憶媒体に記憶させた状態で、ユーザーに提供してもよい。そのパーソナルコンピュータ等が、LAN、インターネット、電話回線等の通信ネットワークに接続されている場合には、通信ネットワークを介して、動作プログラムや各種データ等をパーソナルコンピュータ等に提供してもよい。

【0017】鍵盤2Cは発音すべき楽音の音高を選択するための複数の鍵を備えており、各鍵に対応したキースイッチを有しており、また必要に応じて押圧力検出装置等のタッチ検出手段を有している。鍵盤2Cは音楽演奏のための基本的な操作子であり、これ以外の演奏操作子でもよいことはいうまでもない。押鍵検出回路2Dは発生すべき楽音の音高を指定する鍵盤2Cのそれぞれの鍵に対応して設けられたキースイッチ回路を含むものである。この押鍵検出回路2Dは鍵盤2Cの離鍵状態から押鍵状態への変化を検出してキーオンイベントを出力し、押鍵状態から離鍵状態への変化を検出してキーオフイベ

ントを出力すると共にそれぞれのキーオンイベント及びキーオフイベントに関する鍵の音高を示すノートナンバを出力する。押鍵検出回路2Dはこの他にも鍵押し下げ時の押鍵操作速度や押圧力等を判別してベロシティデータやアフタタッチデータを出力する。

【0018】パネルスイッチ2Eは自動演奏スタート/ストップスイッチ、一時停止（ポーズ）スイッチ、音色、音量、効果等を選択、設定、制御するための各種スイッチを含むものである。スイッチ検出回路2Fはパネルスイッチ2E上の各スイッチ群に対応して設けられており、これらの各スイッチ群の操作状態に応じたスイッチイベントをバス2Mを介してCPU21に出力する。表示回路2HはCPU21の制御状態、設定データの内容等の各種の情報をディスプレイ2Gに表示するものである。この実施の形態では、著作権表示データ、曲名、作曲者情報、作成年月日、歌詞、ニュース文、機種名（ハード名）、IDなどのテキストデータなどを表示する。ディスプレイ2Gは液晶表示パネル（LCD）等から構成され、表示回路2Hによってその表示動作を制御される。

【0019】音源回路2Jは、複数のチャンネルで楽音信号の同時発生が可能であり、CPU21から与えられた楽音制御情報（ノートオン、ノートオフ、ベロシティ、ピッチデータ、音色番号等のMIDIデータ）を入力し、これらの楽音制御情報に基づいた楽音信号を発生する。音源回路2Jにおいて複数チャンネルで楽音信号を同時に発音させる構成としては、1つの回路を時分割で使用することによって複数の発音チャンネルを形成するようなものや、1つの発音チャンネルが1つの回路で構成されるような形式のものであってもよい。また、音源回路2Jにおける楽音信号発生方式はいかなるものを用いてもよい。例えば、発生すべき楽音の音高に対応して変化するアドレスデータに応じて波形メモリに記憶した楽音波形サンプル値データを順次読み出すメモリ読み出し方式（波形メモリ方式）、又は上記アドレスデータを位相角パラメータデータとして所定の周波数変調演算を実行して楽音波形サンプル値データを求めるFM方式、あるいは上記アドレスデータを位相角パラメータデータとして所定の振幅変調演算を実行して楽音波形サンプル値データを求めるAM方式等の公知の方式を適宜採用してもよい。また、これらの方式以外にも、自然楽器の発音原理を模したアルゴリズムにより楽音波形を合成する物理モデル方式、基本波に複数の高調波を加算することで楽音波形を合成する高調波合成方式、特定のスペクトル分布を有するフォルマント波形を用いて楽音波形を合成するフォルマント合成方式、VCO、VCF及びVCAを用いたアナログシンセサイザ方式等を採用してもよい。また、専用のハードウェアを用いて音源回路を構成するものに限らず、DSPとマイクロプログラムを用いて音源回路を構成するようにしてもよいし、CPU

とソフトウェアのプログラムで音源回路を構成するようにしてもよい。

【0020】タイマ2Nは時間間隔を計数したり、自動演奏のテンポを設定したりするためのテンポクロックパルスを発生する。このテンポクロックパルスの周波数は各種スイッチ2Eの中のテンポ設定スイッチや予め演奏データに含まれている図3のようなテンポデータに基づいて決定されている。タイマからのテンポクロックパルスはCPU21に対してインタラプト命令として与えられ、CPU21はインタラプト処理により自動演奏時における各種の処理を実行する。効果回路2Kは音源回路2Jからの楽音信号に種々の効果を付与し、効果の付与された楽音信号をサウンドシステム2Lに出力する。効果回路2Kによって効果の付与された楽音信号は、アンプ及びスピーカからなるサウンドシステム2Lを介して発音される。

【0021】次に、この発明に係る電子情報処理システムが電子情報を付与すると共に暗号化処理を同時に行う場合の動作の一例について説明する。図3は、電子楽器1が電子情報処理システムとして動作する場合のMIDIデータ編集処理の一例を示すフローチャート図である。まず、この場合には、電子楽器1は、フロッピーディスクドライブ24から読み出したMIDIデータにそのヘッダ情報の一部である電子署名すなわち著作権表示データを分散して記憶すると共にMIDIフォーマットを維持したまま暗号化処理を施す場合について説明する。

【0022】まず、ステップ31では、MIDIデータ列内に分散して書き込まれるべき付属情報の内容を決定する。この実施の形態では、付属情報として、電子署名（著作権表示データ）をMIDIデータ列内に分散して埋め込む場合について説明する。例えば、著作権表示データとして『COPYRIGHT△YMH△1996』のような文字をMIDIデータ列内に分散して書き込む場合には、これらの文字列をパネルスイッチ2Eを用いて入力する。ここで、△は空白を意味する。ステップ31で書き込むべき電子署名すなわち付属情報の文字列が決定したので、今度はステップ32で、その電子署名に関する4ビット構成のデータ列を得る。例えば、パネルスイッチ2Eによって『COPYRIGHT△YMH△1996』が入力された場合には、それをASCIIの文字符号のデータ列に変換する。この場合、それぞれの文字は『C』=『43H』、『O』=『4FH』、『P』=『50H』、『Y』=『59H』、『R』=『52H』、『I』=『49H』、『G』=『47H』、『H』=『48H』、『T』=『54H』、『△』=『20H』、『Y』=『59H』、『M』=『4DH』、『H』=『48H』、『△』=『20H』、『1』=『31H』、『9』=『39H』、『9』=『39H』、『6』=『36H』のような一連

のASCIIの4ビット構成のデータで表されるようになる。なお、『36H』のように記された数において、数字の末尾のHはその数が16進表示であることを示す。

【0023】次に、ステップ33では、電子署名として付与する情報すなわち、ステップ32で得られたASCIIの4ビット構成のデータ列を4ビットレジスタBRに格納する。例えば、『Y』=『59H』、『M』=『4DH』、『H』=『48H』の文字をMIDIデータ列に順番に格納する場合、4ビットレジスタBRには、図1(C)に示すように『0101B』、『1001B』、『0100B』、『1101B』、『0100B』、『1000B』のような4ビット構成のデータが順番に格納される。なお、『1000B』のように記された数において、数字の末尾のBはその数が2進表示であることを示す。そして、ステップ34では、この4ビットレジスタBRに格納されている上位3ビットをノートエリアナンバNAとし、最下位ビット(LSB)をベロシティエリアナンバVAとする。従って、電子署名の『Y』=『59H』の最初の4ビット：『0101B』の場合はNA=2, VA=1、後ろの4ビット：『1001B』の場合はNA=4, VA=1となり、電子署名の『M』=『4DH』の最初の4ビット：『0100B』の場合はNA=2, VA=0、後ろの4ビット：『1101B』の場合はNA=6, VA=1となり、電子署名の『H』=『48H』の最初の4ビット：『0100B』の場合はNA=2, VA=0、後ろの4ビット：『1000B』の場合はNA=4, VA=0となる。

【0024】次に、ステップ35では、MIDIデータ列(Standard MIDI File: SMF)の中からキーオンイベントデータやプログラムチェンジデータやコントロールチェンジデータなどの各種のMIDIデータを順次取り出す。すなわち、MIDIデータ列は基本的にはキーオンステータスバイト、キーコードバイト、ベロシティバイトからなるキーオンイベントデータやこれ以外のプログラムチェンジイベントデータやコントロールチェンジイベントデータなどから構成されているので、ステップ35では、このようなMIDIデータを順番に取り出す。

【0025】ステップ36では、取り出されたMIDIデータがキーオンイベントデータKONであるかどうかを判定し、キーオンイベントデータ(YES)の場合は次のステップ37以下の処理を行い、そうでない(NO)の場合はステップ3Hに進む。従って、ステップ35で取り出されたMIDIデータがキーオンイベントデータの場合には、図1(A)のようなキーオンイベントデータからなるMIDIデータ列SMF1が得られることになる。このキーオンイベントデータのMIDIデータ列SMF1はデュレーションタイムΔと、キーオンイ

ベントデータとの組合せで構成される。ステップ37では、キーオンイベントデータの中の各バイトのデータをそれぞれ対応するキーコードレジスタb、ベロシティレジスタcに格納する。すなわち、キーオンイベントデータのキーオンステータスバイト中のキーコードバイトの値すなわちキーコードをキーコードレジスタbに、ベロシティバイト中のベロシティの値をベロシティレジスタcにそれぞれ格納する。

【0026】ステップ38では、前のキーオンイベントデータのキーコードとベロシティとの差分値を取り、それをキーコード差分値レジスタdkey及びベロシティ差分値レジスタdvelにそれぞれ格納する。すなわち、キーコードレジスタbには今回のキーオンイベントデータのキーコードが格納され、前回値レジスタb0には前のキーオンイベントデータのキーコードが格納される。キーコードレジスタbの値から前回値レジスタb0の値を減算することによって得られた差分値がキーコード差分値レジスタdkeyに格納される。ベロシティレジスタcには今回のキーオンイベントデータのベロシティが格納され、前回値レジスタc0には前のキーオンイベントデータのベロシティが格納される。従って、同様にベロシティレジスタcの値から前回値レジスタc0の値を減算することによって得られた差分値がベロシティ差分値レジスタdvelに格納される。図1(A)に示されるMIDIデータ列SMF1の場合、このステップ38の演算によって得られたキーコード差分値レジスタdkey及びベロシティ差分値レジスタdvelの値は図1(B)のようになる。ステップ39では、次のステップ38の処理に備えて、今回のキーコードの値すなわちキーコードレジスタbの値を前回値レジスタb0に格納し、今回のベロシティの値すなわちベロシティレジスタcの値を前回値レジスタc0に格納する。

【0027】ステップ3Aでは、キーコード差分値レジスタdkeyの値が8よりも小さいかどうかを判定し、小さい場合には、次のステップ3Bに進み、8以上の場合にはステップ3Gに進む。ステップ3Bでは、ベロシティ差分値レジスタdvelの値が32よりも小さいかどうかを判定し、小さい場合には、次のステップ3Cに進み、32以上の場合にはステップ3Gに進む。すなわち、両方のステップ3A及び3BでNOと判定された場合には、そのキーオンイベントデータは置換対象外のデータだと判定し、ステップ3Gでレジスタcの最下位ビットを『0』に修正、すなわちベロシティの値を偶数にしてからステップ3Hに進む。これによって、ベロシティの値が偶数の場合には、そのキーオンイベントデータが置換対象外のデータであり、修正されなかったことをデータ修復の際に容易に判定することができるようになる。

【0028】ステップ3Cでは、キーコード差分値レジスタdkeyの値が正又は0の場合と負の場合とで、そ

れぞれ異なる置換演算を行う。すなわち、キーコード差分値レジスタdkeyの値が正又は0の場合には、ノートエリアナンバNAを8倍したものと、キーコード差分値レジスタdkeyの値と、定数64との和を求め、その結果を第1キーコードレジスタb1に格納する。3ビットからなるノートエリアナンバNAの最大値は

“7”、最小値は“0”であるから、この条件下では、レジスタb1の値の最小値は“64”である。一方、キーコード差分値レジスタdkeyの値が負の場合には、定数63からノートエリアナンバNAを8倍したものを減算すると共にキーコード差分値レジスタdkeyの値を加算し、その結果を第1キーコードレジスタb1に格納する。なお、このときキーコード差分値レジスタdkeyは負の値なので、結果的には定数63からキーコード差分値レジスタdkeyの値を減算することになる。この条件下では、レジスタb1の値は“63”よりも大きくなることはない。

【0029】ステップ3Dでは、ベロシティ差分値レジスタdvelの値が正又は0の場合と負の場合とで、それぞれ異なる置換演算を行う。すなわち、ベロシティ差分値レジスタdvelの値が正又は0の場合には、ベロシティエリアナンバVAを32倍したものと、ベロシティ差分値レジスタdvelの値と、定数64との和を求め、その結果を第1ベロシティレジスタc1に格納する。この条件下では、レジスタc1の値の最小値は“64”である。ベロシティ差分値レジスタdvelの値が負の場合には、定数63からベロシティエリアナンバVAを32倍したものを減算すると共にベロシティ差分値レジスタdvelの値を加算し、その結果を第1ベロシティレジスタc1に格納する。なお、このときベロシティ差分値レジスタdvelは負の値なので、結果的には定数63からベロシティ差分値レジスタdvelの値を減算することになる。この条件下では、レジスタc1の値は“63”よりも大きくなることはない。

【0030】ステップ3Eでは、第1キーコードレジスタb1の値を新しいキーコードの値とし、第1ベロシティレジスタc1の値を新しいベロシティの値としてMIDIデータを修正する。そして、ステップ3Fで、第1ベロシティレジスタc1の最下位ビットを『1』に修正、すなわちベロシティの値を奇数に修正してからステップ3Hに進む。これによって、ベロシティの値が奇数の場合には、そのキーオンイベントデータが付属情報に基づいて修正されたことをデータ修復の際に容易に判定することができる。ステップ3Hでは、全ての付属情報についてステップ33からステップ3Gまでの処理が終了したかどうかを判定し、終了している(YES)場合にはこのMIDIデータ編集処理を終了し、終了していない場合にはステップ33にリターンし、次の付属情報について一連の処理を行う。

【0031】例えば、図1の場合には、1番目のキーオ

ンイベントデータ(91, 100, 100)は、最初のものなので修正されない。2番目のキーオンイベントデータ(91, 101, 102)は、図3のMIDIデータ編集処理によって、図1(E)に示すような2番目のキーオンイベントデータ(91, 81, 99)に変換される。同様にして、3番目のキーオンイベントデータ(91, 103, 102)も図1(D)に示すような3番目のキーオンイベントデータ(91, 98, 97)に変換される。そして、4番目のキーオンイベントデータ(91, 120, 70)は、図1(B)に示すように、キーコード差分値レジスタdkeyの絶対値が『17』と8以上であるため、ステップ3AでNOと判定され、置換対象外となり、ステップ3Gを経て、キーオンイベントデータ(91, 120, 70)がそのまま図1(D)に示すように4番目のキーオンイベントデータとなる。そして、5番目から8番目までのキーオンイベントデータも同様にして、図1(D)の5番目から8番目までのキーオンイベントデータのように変換される。なお、この場合に、2番目と3番目のキーオンイベントデータの場合は、ステップ3Fの処理によって、ベロシティの値が奇数に変換されている。なお、プログラムチェンジイベントは付属情報の付与されるノートオンイベントなどに比べて、比較的低頻度で発生し、プログラムチェンジ後は、楽器が変わることを意味するので、その位置を始点としてエディットされる可能性の高い単位ということが言えるので、取り出されたMIDIデータがプログラムチェンジイベントだった場合には、直ちに終了して、付属情報の最初からMIDIデータを編集するようにしてもよい。

【0032】図5は、上述の第1の実施の形態に従って得られる各レジスタb1, c1の値のとり得る状態を列挙したものである。図5において、縦軸にはレジスタb1にストアされるキーコードのとりうる値を割り当て、横軸にはレジスタc1にストアされるベロシティデータのとりうる値を割り当て、各値の63と64との間を原点とするような直交座標系を構成する。これによって、キーオンイベントデータの中のキーコードが高くなる場合(正の値)又は小さくなる場合(負の値)と、ベロシティが増加する場合(正の値)又は減少する場合(負の値)とからなる4種類の組み合わせがこの直交座標系の各象限に割り当てられることになる。そして、各象限には付属情報となる4ビット構成の16個のデータが割り当てられる。すなわち、図5に示すようにキーコードが高くなり、ベロシティが増加する場合と、キーコードが高くなり、ベロシティが減少する場合と、キーコードが低くなり、ベロシティが増加する場合と、キーコードが低くなり、ベロシティが減少する場合のそれぞれが各象限に対応するようになる。各象限に割り当てられた4ビット構成の付属情報のうち、最下位ビットは横軸のベロシティに、上位3ビットは縦軸のキーコードに割り当て

られる。従って、キーコードが高くなり、ベロシティが増加する場合には、直交座標の第1象限に割り当てられ、キーコード軸の64~127は000から111までの8つの付属情報に対応するように分割され、ベロシティ軸の64~127は0と1の2つの付属情報に対応するように分割される。キーコードが高くなり、ベロシティが減少する場合には、直交座標の第2象限に割り当てられ、キーコード軸の64~127は000から111までの8つの付属情報に対応するように分割され、ベロシティ軸の63~0は0と1の2つの付属情報に対応するように分割される。キーコードが低くなり、ベロシティが減少する場合には、直交座標の第3象限に割り当てられ、キーコード軸の63~0は000から111までの8つの付属情報に対応するように分割され、ベロシティ軸の63~0は0と1の2つの付属情報に対応するように分割される。キーコードが低くなり、ベロシティが増加する場合には、直交座標の第4象限に割り当てられ、キーコード軸の63~0は000から111までの8つの付属情報に対応するように分割され、ベロシティ軸の64~127は0と1の2つの付属情報に対応するように分割される。

【0033】この場合、図6に示すように、各付属情報に対応したキーコードには8個の値が存在し、ベロシティには32個の値が存在するので、ステップ3A及び3Bのように、キーコード差分値レジスタdkeyの値は8よりも小さくしなければならず、ベロシティ差分値レジスタdve1の値は32よりも小さくしなければならないという条件が入っているのである。従って、キーコードが8よりも小さな値で増加又は減少し、ベロシティが32よりも小さな値で増加又は減少した場合には、図5の各象限の付属情報が選択され、その付属情報の値、キーコード差分値及びベロシティ差分値に応じて、元のキーコード及びベロシティが編集される。なお、この場合、ベロシティについては、編集された値であるかどうかを示すために、ステップ3G及びステップ3Fによって最下位ビットに処理が加えられ、編集されたベロシティの値は、最終的には奇数値の16個だけとなる。

【0034】次に、図3のMIDIデータ編集処理によって付与された電子情報すなわち電子署名データを検出、すなわち修復するMIDIデータ修復処理について説明する。図4は、電子楽器1がMIDIデータ修復装置として動作する場合の一例を示すフローチャート図である。この場合には、フロッピーディスクドライブ25から読み出されたMIDIデータに前述のようなヘッダ情報の一部(電子署名すなわち著作権表示データ)が分散記録されているものとする。以下の実施の形態では、この著作権表示データを検出して、それをモニタなどに表示するまでの動作を説明する。まず、ステップ41では、電子署名を検出すべきMIDIデータ列、すなわち図3の電子情報付与処理によって電子署名の付与された

MIDIデータ列(Standard MIDI File: SMF)の中から順次キーオンイベントデータやプログラムチェンジデータやコントロールチェンジデータなどのデータを取り出す。

【0035】ステップ42では、取り出されたMIDIデータがキーオンイベントデータKONであるかどうかを判定し、キーオンイベントデータ(YES)の場合は次のステップ43に進み、そうでない(NO)場合はステップ4Cに進む。従って、ステップ41で取り出されたMIDIデータがキーオンイベントデータの場合には、図1(F)のようなキーオンイベントMIDIデータ列SMF2が作成されることになる。ステップ43では、キーオンイベントデータの中の各バイトのデータをそれぞれ対応するキーコードレジスタb、ベロシティレジスタcに格納する。すなわち、キーコードバイトの中のキーコードをキーコードレジスタbに、ベロシティバイトの中のベロシティをベロシティレジスタcにそれぞれ格納する。

【0036】ステップ44では、ベロシティレジスタcの最下位ビットが『1』かどうか、すなわち、付属情報(電子署名情報)の付与されたキーオンイベントデータであるかどうかの判定を行い、YESの場合は次のステップ45に進み、そうでない(NO)場合はステップ4Cに進む。ステップ45、ステップ46、ステップ49及びステップ4Aでは、相前後するキーオンイベントデータのキーコードレジスタb、前回キーコードレジスタb0、ベロシティレジスタc及び前回ベロシティレジスタc0のそれぞれの格納値に基づいて修復キーコードレジスタb2、修復ベロシティレジスタc2、ノートエリアナンバNA及びベロシティエリアナンバVAの値をそれぞれ求める。このとき、キーコードレジスタbの値が64以上の場合とこれよりも小さい場合とで、それぞれ異なる演算式を用いて各値を求める。

【0037】ステップ45においてはステップ3C(図3)の置換式の逆算を行なう。すなわち、キーコードレジスタbの値が64以上の場合には、キーコードレジスタbの値から定数64を減算し、その減算値の8におけるモジュロ(減算値を8で除した余り)を求めることで、dkeyを再生する。そして、このdkeyを前回キーコードレジスタb0に加算することでキーコードの修復値を得て、それを修復キーコードレジスタb2にストアする。一方、キーコードレジスタbの値が64よりも小さい場合には、キーコードレジスタbの値から定数63を減算し、その減算値の8におけるモジュロを求めることで、dkeyを再生する。そして、このdkeyを前回キーコードレジスタb0に加算することでキーコードの修復値を得て、それを修復キーコードレジスタb2にストアする。

【0038】ステップ46においてはステップ3D(図3)の置換式の逆算を行なう。すなわち、ベロシティレ

ジスタcの値が64以上の場合には、ベロシティレジスタcの値から定数64を減算し、その減算値の32におけるモジュロ(減算値を32で除した余り)を求めることで、dvelを再生する。そして、このdvelを前回ベロシティレジスタc0に加算することでベロシティデータの修復値を得て、それを修復ベロシティレジスタc2にストアする。一方、ベロシティレジスタcの値が64よりも小さい場合には、ベロシティレジスタcの値から定数63を減算し、その減算値の32におけるモジュロを求めることで、dvelを再生する。そして、このdvelを前回ベロシティレジスタc0に加算することでベロシティデータの修復値を得て、それを修復ベロシティレジスタc2にストアする。

【0039】ステップ47において、修復キーコードレジスタb2の値をキーコードの値とし、修復ベロシティレジスタc2の値をベロシティの値としてMIDIデータを修復する。ステップ48では、次のステップ45及びステップ46の処理に備えて、修復されたキーコードの値すなわち修復キーコードレジスタb2の値を前回値レジスタb0に格納し、修復されたベロシティの値すなわち修復ベロシティレジスタc2の値を前回値レジスタc0に格納する。ステップ49においてはステップ3C(図3)の置換式の逆算を行なうことでNAを再生する。すなわち、キーコードレジスタbの値が64以上の場合には、キーコードレジスタbの値から定数64を減算し、その減算値を8で除した商を求め、それをノートエリアナンバNAとする。一方、キーコードレジスタbの値が64よりも小さい場合には、定数63からキーコードレジスタbの値を減算し、その減算値を8で除した商を求め、それノートエリアナンバNAとする。ステップ4Aではステップ3D(図3)の置換式の逆算を行なうことでNAを再生する。すなわち、ベロシティレジスタcの値が64以上の場合には、ベロシティレジスタcの値から定数64を減算し、その減算値を32で除した商を求め、それをベロシティエリアナンバVAとする。一方、ベロシティレジスタcの値が64よりも小さい場合には、定数63からベロシティレジスタcの値を減算し、その減算値を32で除した商を求め、それベロシティエリアナンバVAとする。

【0040】ステップ4Bでは、ノートエリアナンバNAを上位3ビットとし、ベロシティエリアナンバVAを最下位ビットとする4ビット構成のデータ列からなる付属情報を得る。ステップ4Cでは、全MIDIデータの修復が終了したかどうかを判定し、終了した場合にはMIDIデータ修復処理を終了し、そうでない場合にはステップ41にリターンし、再びMIDIデータに対する修復処理を行う。以上の一連の修復処理が終了した時点でモニタにそのヘッダ情報の一部(電子署名すなわち著作権表示データ)などを表示するが、ここではそのステップについて省略してある。

【0041】例えば、図1(E)のキーオンイベントのMIDIデータ列SMF2と図1(G)のキーオンイベントのMIDIデータ列SMF2とは同じものであり、図1(G)のキーオンイベントのMIDIデータ列SMF2に基づいて、図1(F)のようなMIDIデータ列SMF3が修復され、図1(G)のような付属情報が抽出される。まず、図1(G)の1番目のキーオンイベントデータ(91, 100, 100)は、ベロシティの最下位ビットが『0』すなわち偶数なのでそのままの値がキーオンイベントデータとなる。次に、2番目のキーオンイベントデータ(91, 81, 99)は、ベロシティの最下位ビットが『1』すなわち奇数であり、64以上なので、図4のMIDIデータ修復処理によって、図1(F)に示すような2番目のキーオンイベントデータ(91, 101, 103)に変換される。同様に、3番目のキーオンイベントデータ(91, 98, 97)も図1(F)に示すような3番目のキーオンイベントデータ(91, 103, 104)に変換される。そして、4番目のキーオンイベントデータ(91, 120, 70)は、ベロシティの最下位ビットが『0』すなわち偶数なのでそのままの値がキーオンイベントデータとなる。そして、図1(G)に示すような5番目から8番目までのキーオンイベントデータも同様に、図1(F)の5番目から8番目までのキーオンイベントデータのように変換される。この場合に、2番目と3番目のキーオンイベントデータについては、図3のステップ3Fの処理によってベロシティの値が奇数に変換されている関係上、図1(F)に示されるキーオンイベントデータのベロシティの値がMIDIデータ編集処理される前の図1(A)のキーオンイベントデータのベロシティの値よりも1~2程度大きくなっている。しかしながら、この程度のベロシティ値の誤差は人間の耳に与える影響は小さいので、ほとんど無視できる範囲の誤差であると言える。また、図1(G)のキーオンイベントデータに基づいて、図1(H)のような電子署名情報(著作権表示データ)が再現されることになる。このようにして修復された電子署名情報を図示していない表示装置などで表示する。

【0042】なお、上述の第1の実施の形態では、ベロシティに1ビット、キーコードに3ビットの付属情報を割り当てる場合について説明したが、これ以外のビットの組み合わせでもよいことはいうまでもない。例えば、ベロシティとキーコードにそれぞれ2ビットの付属情報を割り当ててもよいし、ベロシティのみに4ビットの付属情報を割り当ててもよい。上述の第1の実施の形態では、ノート方向を8等分、ベロシティ方向を2等分して4ビット構成の付加情報を記録する場合について説明したが、ノート方向及びベロシティ方向の分割数を任意に設定することにより、付加情報量を増やすことが可能である。例えば、ノート、ベロシティの両方とも16等分

すれば、1キーオンイベントに8ビット構成の付加情報を付加することもでき、アスキーコードの1バイトならば1イベントのキーオンで表現することも可能となる。なお、この場合には、キーコード差分値レジスタ`dkey`及びベロシティ差分値レジスタ`dvel`の値が4以下でないと、そのキーオンイベントには付加情報を割り当てることができなくなるという制限が生じる。従って、予めキーコード差分値及びベロシティ差分値の発生頻度を取得し、それに応じて最も沢山の付属情報を埋め込むことが可能な分割数を選択するようにしてもよい。なお、この場合、暗号化のためのアルゴリズムも、1データ単位に組み込むべき付属情報のドット数に応じて、幾分変更される。例えば、4ビットの付属情報をノートデータに組み込む場合は、図3のステップ3Cでの暗号化アルゴリズムの係数“8”が“4”に変更となる。また、上述の第1の実施の形態では、図5に示すような関係を演算式によって形成する場合について説明したが、予め図5のような関係となるようなテーブルを作成しておき、各値をテーブル変換することによって、所定の値が求まるようにしてもよい。この場合には、図5のように付属情報が規則正しく並んでいなくても、ランダムに配列してあってもよいことになる。なお、テーブルは、ROMやRAMのようなメモリ素子で構成するものに限らず、ロジックゲート回路で構成してもよいし、ソフトウェアプログラムで構成してもよい。

【0043】上述の第1の実施の形態では、フロッピーディスクドライブ24から読み出されたMIDIデータにおいてそのヘッダ情報の一部である電子署名すなわち著作権表示データを比較的大きなデータ単位でキーオンイベントデータ中に分散して記憶すると共にMIDIフォーマットを維持したまま暗号化処理を施す(電子透しを施す)という場合について説明した。これだと、キーオンイベントデータの中にはそのキーコード及びベロシティの内容がほとんど書き換えられるものが出てくるので、所定のMIDIデータ修復処理を行わずにMIDIデータ再生処理を行っても再生される楽音は、全くデータラメなものになってしまう。この点に鑑みて、次に説明する第2の実施の形態では、上記とは異なるやり方で

「電子透し」を施すようにしている。この第2の実施の形態では、MIDIデータの最初の部分に、複数の暗号化処理の中のどの暗号化処理が施されたのかを示すスクランブルデコードデータを埋め込み、そこには暗号化処理を行わずに、通常のMIDIデータの再生を行えるようにし、残りのMIDIデータ部分にスクランブルデコードデータに対応したスクランブル処理を行い、MIDIデータの再生を行うことができないようにしている。

【0044】図7は、MIDIデータ中に埋め込まれるスクランブルデコードデータの内容を示す図である。このスクランブルデコードデータは2バイト構成のデータであり、第1バイト目の上位4ビットはそのアルゴリズム

ムを示し、下位4ビットはスクランブル処理されるイベント数すなわちカウント数を示し、第2バイト目の8ビットはそのアルゴリズムに対応したバリューを示す。ここで、アルゴリズムの種類としては、キーオンデータのノート番号を変更するというもの、キーオンデータのノートとベロシティを入れ替えるというもの、インターバルデータを変更するというもの、キーオンデータのチャンネルデータを変更するというものなどがある。カウント数は、前述のようにASCIIの2文字分すなわち16ビット分のデータを作成するのに必要な数のキーオンイベントデータの先頭からいくつに対してスクランブル処理を行うのかを示すものである。

【0045】上述の第1の実施の形態の場合には、1つのキーオンイベントデータに4ビット分のデータを埋め込むことができたので、2つのキーオンイベントデータにASCIIの1文字分の付属情報を埋め込むことができたが、今回の第2の実施の形態では、1つのキーオンイベントデータに1ビット分のデータしか埋め込めない場合について説明するので、修正するイベント数の最大値は16となる。バリューは対応するアルゴリズムによって異なるものであり、例えば、キーオンデータのノート番号を変更するというアルゴリズムの場合には、その変更する度合いすなわちトランスポーズ値(修正ノート値)であり、インターバルデータを変更するというアルゴリズムの場合には、その変更する修正値であり、キーオンデータのチャンネルデータを変更する場合には、その修正チャンネル値である。このように全部で16種類のアルゴリズムが規定されているものとする。なお、この16種類のアルゴリズムの中の一つとして、前述の第1の実施の形態に係るようなスクランブル処理が存在してもよいことはいうまでもない。

【0046】図8は、電子楽器1が電子情報処理システムとして動作する場合のMIDI編集処理の別の一例となるMIDI編集処理2を示すフローチャート図である。まず、この場合には、電子楽器1は、フロッピーディスクドライブ24から読み出したMIDIデータにそのヘッダ情報の一部である電子署名すなわち著作権表示データを分散して記憶すると共に前述の16ビット構成のスクランブルデコードデータを記憶する。まず、ステップ81では、MIDIデータ列内に分散して書き込まれるべきスクランブルの内容を決定する。すなわち、図8のどのアルゴリズムに基づいてスクランブル処理を行うのがパネルスイッチ2Eによって予め設定されるものとする。スクランブルの内容がステップ81で決定したので、今度はステップ82でそのスクランブルのデコードデータ列を得る。例えば、パネルスイッチ2Eによって、第5番目のアルゴリズムが選択された場合には、そのアルゴリズムの種類を示すデータ『0101B』と、そのアルゴリズムによって修正されるカウント数『1001B』とからなる8ビット構成のデータ列がス

クランブルデコードデータ列としてMIDIデータ列内に分散して記録されるようになる。

【0047】次に、ステップ83では、ステップ82で得られたスクランブルデコードデータ列の1バイト分をバイトレジスタBRに格納する。そして、ステップ84では、このバイトレジスタBRに格納されたスクランブルデコードデータの各ビットを反転して、別の8ビット構成のビット列を作成する。例えば、スクランブルデコードデータ列が図12(F)に示すように『01011001B』のような場合には、ステップ84のデータ変換すなわちビット反転によって、図12(E)のようなビット列『10100110B』になる。次に、ステップ85では、MIDIデータ列(Standard MIDI File: SMF)の中からキーオンイベントデータやプログラムチェンジデータやコントロールチェンジデータなどの各種のMIDIデータを順次取り出す。すなわち、MIDIデータ列は基本的にはキーオンステータスバイト、キーコードバイト、ベロシティバイトからなるキーオンイベントデータやこれ以外のプログラムチェンジイベントデータやコントロールチェンジイベントデータなどから構成されているので、ステップ85では、このようなMIDIデータを順番に取り出す。

【0048】ステップ86では、取り出されたMIDIデータがキーオンイベントデータKONであるかどうかを判定し、キーオンイベントデータ(YES)の場合は次のステップ87に進み、そうでない(NO)の場合はステップ8Dに進む。従って、ステップ85で取り出されたMIDIデータがキーオンイベントデータの場合には、図12(A)のようなキーオンイベントデータからなるMIDIデータ列SMF1が得られることになる。このキーオンイベントデータのMIDIデータ列SMF1はデュレーションタイムDと、キーオンイベントデータとの組合せで構成される。ステップ87では、キーオンイベントデータの中の各バイトのデータをそれぞれ対応するレジスタa, b, cに格納する。すなわち、キーオンイベントデータの中のキーオンステータスバイトの中のチャンネル番号をチャンネルレジスタaに、次の第1のデータバイト(キーコードバイト)の中のキーコードをキーコードレジスタbに、第2のデータバイト(ベロシティバイト)の中のベロシティをベロシティレジスタcにそれぞれ格納する。

【0049】ステップ88では、ステップ84で作成されたビット列の先頭から1ビット取り出して、それを第1のビットフラグBF1に格納する。次のステップ89では、ステップ87の各レジスタa, b, cの格納値を所定の関数に従って演算し、その演算結果によって得られたビットデータを反転して、第2のビットフラグBF2に格納する。この実施の形態では、各レジスタa, b, cの値の合計値のモジュロ2を所定の関数とする。すなわち、関数 $f1(a, b, c) = (a + b + c)$

mod 2とする。例えば、図12(A)のようなキーオンイベントMIDIデータ列SMF1の場合には、8つのキーオンイベントデータでスクランブルデコードデータのアルゴリズム及びカウント数に相当するデータが作成される。まず、各キーオンイベントデータの所定の関数 $f1(a, b, c)$ で演算すると、図12(B)のようになる。すなわち、 $(a+b+c)$ の値が偶数なら『0』、奇数なら『1』となる。そして、これらの反転ビットが第2のビットフラグBF2に格納され、図12(C)のようになる。

【0050】ステップ8Aでは、ステップ88及びステップ89で得られた第1のビットフラグBF1と第2のビットフラグBF2とが等しいかどうかを判定し、等しい(YES)場合には次のステップ8B及びステップ8Cの処理を行い、等しくない(NO)場合はステップ8Eにジャンプする。ステップ8Bでは、第1のビットフラグBF1と第2のビットフラグBF2とが等しいと判定されたので、ベロシティレジスタcの値すなわちベロシティデータの最下位ビットに『1』を加算する。そして、ステップ8Cで、そのレジスタcの値に基づいてMIDIデータ列中のキーオンイベントデータを構成するベロシティの値を更新する。すなわち、ベロシティの値を1だけ増加する。例えば、図12の場合には、第1のビットフラグBF1と第2のビットフラグBF2との3番目、5番目、6番目及び7番目がそれぞれ等しいと、ステップ8Aで判定されるので、MIDIデータ列中の3番目、5番目、6番目及び7番目のキーオンイベントデータのベロシティの値が『1』だけ増加される。すなわち、3番目のキーオンイベントデータのベロシティ値『63』が『64』になり、5番目のキーオンイベントデータのベロシティ値『78』が『79』になり、6番目のキーオンイベントデータのベロシティ値『91』が『92』になり、7番目のキーオンイベントデータのベロシティ値『42』が『43』になる。

【0051】ステップ8Dでは、ステップ86又はステップ8AでNOと判定された場合又はステップ8Cの処理を終えた場合に行われる処理であって、ステップ85～ステップ8Cまでの処理を1巡回処理とした場合にこの1巡回処理が8回分すなわち8ビット分行われたかどうかを判定し、8回分行われた(YES)場合には、次のステップ8Eに進み、そうでない(NO)の場合は、ステップ85にリターンし、次の1ビットに対してステップ85～ステップ8Cの処理を行う。ステップ8Eでは、ステップ83～ステップ8Dまでの処理を1巡回処理とした場合にその1巡回処理がスクランブルデコードデータの構成バイト数だけ行われたかどうかを判定し、行われた(YES)場合には、次のステップ8Fに進み、そうでない(NO)場合は、ステップ83にリターンし、スクランブルデコードデータの次の1バイトのバリュースに対してステップ83～ステップ8Dの処理を行

う。

【0052】ステップ8Fでは、ステップ81で選択されたスクランブルの内容に応じて、MIDIデータ列の後半部分にスクランブル処理を施す。例えば、前述のスクランブルデコードデータの埋め込み終了後のキーオンデータに対してスクランブル処理を行うようにしてもよいし、スクランブルデコードデータの埋め込み終了後、所定数のキーオンデータ経過後にスクランブル処理を行うようにしてもよい。また、スクランブルデコードデータの埋め込みを複数回行い、その後にスクランブル処理を行うようにしてもよい。この場合に、取り出されたMIDIデータがプログラムチェンジイベントPCMの場合には、スクランブルの内容をランダムに変更して、スクランブルデコードデータの埋め込み処理及びスクランブルの内容に応じたMIDIデータ列の編集処理を行うようにしてもよい。プログラムチェンジイベントはノートオンイベントなどに比べて、比較的低頻度で発生し、プログラムチェンジ後は、楽器が変わることを意味するので、その位置を始点としてスクランブルの内容を変更し、そのスクランブルデコードデータを埋め込み、スクランブル処理を行うことによって、一定の楽器についてはスクランブルの掛からない演奏を行うことができ、ある所定時間経過後に演奏にスクランブルをかけることができる。

【0053】次に、図8のMIDI編集処理2によって付与されたスクランブルデコードデータの修復処理について説明する。なお、スクランブル処理の解除については、アルゴリズムに応じて逆の処理を施せばいいので、ここではその説明は省略する。図9は、電子楽器1がMIDIデータ修復装置として動作する場合の一例となるMIDIデータ修復処理2を示すフローチャート図である。この場合には、フロッピーディスクドライブ24から読み出されたMIDIデータのベロシティバイト部分にそのヘッダ情報の一部である電子署名すなわち著作権表示データが分散記録されているとともにスクランブルデコードデータが記録されているものとする。以下の実施の形態では、このスクランブルデコードデータを検出して、それに基づいてMIDIデータを修復するまでの動作について説明する。

【0054】まず、ステップ91では、デコードデータ格納レジスタDECODEにナルデータを格納する。すなわち、デコードデータ格納レジスタDECODEの内容をリセットする。ステップ92では、スクランブルデコードデータを検出すべきMIDIデータ列、すなわち図8のMIDIデータ編集によってスクランブルデコードデータの付与されたMIDIデータ列を取り出す。そして、次のステップ93で、バイトレジスタBRとビットカウンタBCNの値をリセットする。

【0055】次に、ステップ94では、MIDIデータ列(Standard MIDI File: SMF)の

中から順次キーオンイベントデータやプログラムチェンジデータやコントロールチェンジデータなどのデータを取り出す。ステップ95では、取り出されたMIDIデータがキーオンイベントデータKONであるかどうかを判定し、キーオンイベントデータ(YES)の場合は次のステップ96に進み、そうでない(NO)場合はステップ9Cに進む。従って、ステップ94で取り出されたMIDIデータがキーオンイベントデータの場合には、図12(G)のようなキーオンイベントMIDIデータ列SMF2が作成されることになる。なお、図12

(G)のキーオンイベントデータ列は先に電子署名の付与された図12(D)のキーオンイベントデータ列SMF2と同じものである。

【0056】ステップ96では、キーオンイベントデータの中の各バイトのデータをそれぞれ対応するレジスタa, b, cに格納する。すなわち、キーオンイベントデータの中のキーオンステータスバイトの中のチャンネル番号をチャンネルレジスタaに、次の第1のデータバイト(キーコードバイト)の中のキーコードをキーコードレジスタbに、第2のデータバイト(ベロシティバイト)の中のベロシティをベロシティレジスタcにそれぞれ格納する。ステップ97では、バイトレジスタBRの値を2倍、すなわち、シフトして、その最下位ビットに関数 $f1(a, b, c) = (a + b + c) \bmod 2$ の反転値を加算する。ステップ98では、ビットカウンタBCNの値を1だけインクリメントする。ステップ99では、ビットカウンタBCNの値が『8』になったかどうかを判定し、YESの場合は次のステップ9Aに進み、NOの場合はステップ9Cにジャンプする。

【0057】例えば、図12(G)のようなキーオンイベントのMIDIデータ列SMF2の場合には、8つのキーオンイベントデータで1バイト分のスクランブルデコードデータが作成される。まず、各キーオンイベントデータの所定の関数 $f1(a, b, c)$ で演算すると、図12(H)のようになる。すなわち、 $(a + b + c)$ の値が偶数なら『0』、奇数なら『1』となる。そして、これらの反転ビットは図12(J)のようになり、その値がバイトレジスタBRを順次シフトしていくため、最終的には図12(J)のような1バイト分のスクランブルデコードデータが形成される。このようにして、得られた図12(J)のデータは図12(F)と同じである。すなわち、MIDIデータのベロシティバイトに分散して書き込まれたスクランブルデコードデータ、すなわち図12(F)のようなデータが図12(J)のように忠実に再現されたことになる。

【0058】ステップ9Aでは、バイトレジスタBRの格納値をデコードデータ格納レジスタDECODEに追加記憶する。ステップ9Bで、バイトレジスタBRとビットカウンタBCNの値をリセットする。ステップ9Cでは、ステップ95でNOと判定された場合、又はス

テップ9Bの処理を終了した場合に行われる処理であって、ステップ94～ステップ9Bの処理がスクランブルデコードデータに対応する2バイト分終了したかどうかの判定を行い、この判定結果が2バイト分終了(YES)の場合には、次のステップ9Dに進み、終了していない(NO)場合にはステップ94にリターンし、一連の処理を繰り返し実行する。ステップ9Dでは、スクランブルデコードデータの内容に基づいてスクランブル処理されたMIDIデータの修復処理を行う。

【0059】なお、上述の実施の形態では、MIDIデータ列からキーオンイベントデータを順番に取り出して、それに対して順番にスクランブルデコードデータを格納する場合について説明したが、これだと、1つのMIDIデータ列に対して1種類のスクランブル処理しか行えない。そこで、チャンネル単位毎に異なるスクランブル処理を行えるようにした第3の実施の形態について説明する。図10は、電子楽器1がMIDIデータ編集装置して動作する場合の一例となるMIDIデータ編集処理3を示すフローチャート図である。まず、この場合は、電子楽器1は、フロッピーディスクドライブ24から読み出したMIDIデータ中に前述のようなスクランブルデコードデータと共にヘッダ情報の一部である電子署名すなわち著作権表示データを併せて分散記録する。

【0060】まず、ステップ101では、MIDIデータ列内に分散して書き込まれるべき電子署名(著作権表示データ)の内容を決定するとともに、各チャンネルに対応したスクランブルデコードデータの内容を決定する。例えば、著作権表示データとして『COPYRIGHT△YMH△1996』のような文字をMIDIデータ列内に分散して書き込む場合には、これらの文字列をパネルスイッチ2Eを用いて決定する。ここで、△は空白を意味するものとする。書き込むべき電子署名すなわち文字列及びスクランブルデコードデータの内容が決定したので、今度はステップ32で、その電子署名及びスクランブルデコードデータに関するデータ列を得る。例えば、パネルスイッチ2Eによって『COPYRIGHT△YMH△1996』が入力された場合には、それをASCIIの文字符号のデータ列に変換する。この場合は、『C』=『43H』、『O』=『4FH』、『P』=『50H』、『Y』=『59H』、『R』=『52H』、『I』=『49H』、『G』=『47H』、『H』=『48H』、『T』=『54H』、『△』=『20H』、『Y』=『59H』、『M』=『4DH』、『H』=『48H』、『△』=『20H』、『1』=『31H』、『9』=『39H』、『9』=『39H』、『6』=『36H』からなる一連のASCIIのデータ列が得られることになる。

【0061】これらの文字符号のデータ列は前述の第2の実施の形態において説明した、スクランブルデコードデータと同様の手法でMIDIデータの中に埋め込むこ

とができることはいうまでもない。従って、以下の処理では、これらのASCIIのデータ列をスクランブルデコードデータ列の埋め込み法と同様のやり方で、MIDIデータ列中に分散記録するものとする。すなわち、次のステップ103では、図8のステップ83～ステップ8Fと同様の処理をそのチャンネルに対してそれぞれ行う。このとき、ステップ89の所定の関数に従った演算処理をそのチャンネルに対してそれぞれ異ならせて行う。例えば、ステップ89では、関数 $f1(a, b, c) = (a + b + c) \bmod 2$ を第2のビットフラグBF2に格納しており、これを第1の関数処理とする。そして、この第1の関数処理に加えて、次の第2、第3及び第4の関数処理の中から、適当なものを各チャンネル毎に使い分けるようにする。第2の関数処理は、 $f2(a, b, c) = (a + c) \bmod 2$ 、第3の関数処理は、 $f3(a, b, c) = (b + c) \bmod 2$ 、第4の関数処理は、 $f4(a, b, c) = (c) \bmod 2$ 、である。

【0062】ステップ103では、これらの第1から第4までの関数処理をMIDIチャンネル毎に適宜選択的に切り換えて、図8のステップ83～ステップ8Fと同様の処理を行い、それぞれのチャンネル毎にその選択された関数処理に従って変換された電子署名データ及びスクランブルデコードデータをMIDIデータ中に冗長的に組み込む。このように、チャンネル毎に関数処理の内容を変更することによって、或るチャンネルのデータを一度に変更するようなエディット処理、例えばチャンネル情報の入れ替えやキーコードのシフトなどといったものが行われた場合でも、エディットによって変更される特定のチャンネルのデータに関係しない関数を利用して別のチャンネルのデータから電子署名データやスクランブルデコードデータを復元することができる。

【0063】次に、図10のMIDIデータ編集処理3によって編集されたMIDIデータを修復するMIDIデータ修復処理3について図11を用いて説明する。図11は、電子楽器1がMIDIデータ修復装置として動作する場合の一例であるMIDIデータ修復処理3を示すフローチャート図である。この場合、フロッピーディスクドライブ24から読み出されたMIDIデータの各チャンネル毎に所定の関数処理（前述の第1から第4までのいずれかの関数処理）によって、ベロシティバイト部分にそのヘッダ情報の一部である電子署名（著作権表示データ）及びスクランブルデコードデータが分散記録されているものとする。従って、以下の実施の形態では、分散記録された著作権表示データをチャンネル毎に所定の関数処理によって検出して、それをモニタなどに表示したり、スクランブルデコードデータに基づいてMIDIデータを修復するという動作を行う場合について説明する。

【0064】まず、ステップ111では、電子署名を検

出すべきMIDIデータ列、すなわち図10のMIDIデータ編集処理3によって各チャンネル毎に所定の関数処理に応じて電子署名データ及びスクランブルデコードデータの付与されたMIDIデータ列を取り出す。そして、次のステップ112で、チャンネルカウンタをリセットし、ステップ113で関数カウンタをリセットする。このチャンネルカウンタと関数カウンタは、後の処理で、各チャンネルに対して各関数処理を施すために利用されるものである。例えば、MIDIチャンネルが16チャンネル相当の場合には、チャンネルカウンタは0から15までを巡回的にカウントするように処理される。また、関数カウンタは、前述のように関数処理が4種類の場合には、0～3までを巡回的にカウントするように処理される。関数カウンタが『0』の場合は第1の関数処理 $f1(a, b, c) = (a + b + c) \bmod 2$ を、『1』の場合は第2の関数処理 $f2(a, b, c) = (a + c) \bmod 2$ を、『2』の場合は第3の関数処理 $f3(a, b, c) = (b + c) \bmod 2$ を、『3』の場合は第4の関数処理 $f4(a, b, c) = (c) \bmod 2$ を意味する。

【0065】ステップ114では、チャンネルカウンタの値に該当するチャンネルについて、関数カウンタの値に該当する関数処理を用いて図9のステップ91からステップ9Cまでと同様にして電子署名データ及びスクランブルデコードデータの検出処理を行う。ステップ115では、検出された電子署名データ列に有意な部分があるかどうかの判定を行う。ここで有意な部分とは、ASCIIの文字データによって構成される『COPYRIGHT』なる署名データの一部分である。従って、ステップ115では、図9の処理によってデコードデータ格納レジスタDECODEに順次追加されたASCIIの文字データによって構成された文字列の中に『COPYRIGHT』なる文字列データが存在するかどうかを判定することになる。ステップ115の判定の結果、デコードデータ格納レジスタDECODE内に有意な部分が存在しない（NO）と判定された場合には、次のステップ116に進み、関数カウンタを1だけインクリメントする。そして、次のステップ117で、全ての関数処理に対してステップ114及びステップ115の処理すなわち署名データ及びスクランブルデコードデータの抽出処理が行われたかどうかの判定を行い、行われた（YES）場合には次のステップ118に進み、そうでない（NO）場合はステップ114にリターンし、次の関数処理についてステップ114及びステップ115の処理を行う。逆に、ステップ115の判定の結果、デコードデータ格納レジスタDECODE内に有意な部分が存在した場合には、ステップ119に進み、スクランブルデコードデータの内容に基づいてMIDIデータの修復を行う。すなわち、ステップ115における判定がYESであったということは、検出されたスクランブルデコー

ドデータも有意なものだと判断できるので、そのスクランブルデコードデータに基づいて元のMIDIデータを修復する。なお、検出された電子署名データをモニタ上に表示するようにしてもよいことはいうまでもない。

【0066】ステップ117でYESと判定されたということは、全てのMIDIチャンネルに対して、全ての関数処理を用いて電子署名データ及びスクランブルデコードデータの抽出処理を行った結果、データ列に有意な部分が存在しなかったということだから、ステップ118では、その結果として『電子署名データ及びスクランブルデコードデータの検出失敗』などの文字をモニタ上に表示する。ステップ11Aでは、チャンネルカウンタを1だけインクリメントする。そして、次のステップ11Bで全てのチャンネルに対してステップ114及びステップ115の処理すなわち電子署名データ及びスクランブルデコードデータの抽出処理が行われたかどうかの判定を行い、行われた（YES）場合には処理を終了し、そうでない（NO）場合はステップ113にリターンし、次のMIDIチャンネルについてステップ113～ステップ115の処理を行う。

【0067】なお、MIDIデータに基づく楽音再生演奏において、部分的に変更されたベロシティデータがそのまま演奏に利用されるようにしてよい。これは、ベロシティデータの最下位ビットの値が1だけ増加したとしても、楽音再生精度にはさほどの悪影響を与えないからである。勿論、電子署名データの組込みによって引き起こされたどんな誤差でもベロシティデータから取り除くように、部分的に変更されたベロシティデータを正確な値で再生するようにすることも可能である。なお、上述の実施の形態では、署名データやスクランブルデコードデータをMIDIデータのベロシティ部分に分散記録する場合について説明したが、これは一例であり、これ以外の著作権名、曲の題名、画像／映像の題名などに関する文字データやデータ形式に関するデータやその他の種々のデータを分散記録してもよいことはいうまでもない。

【0068】また、ベロシティ部分以外の、デュレーションタイム（待ち時間）データに分散記録してもよいし、その両方に分散記録するようにしてもよい。両方に分散記録する場合、その厳密性すなわち分散記録することによってそのデータの内容が変更する度合いに応じて、分散記録する箇所を適宜切り換えるようにしてもよい。すなわち、デュレーションタイムデータの短い領域で、そのデュレーションタイムデータに分散記録すると、それによる変動の割合（厳密性）が大きくなるので、好ましくないが、デュレーションタイムデータが比較的大きい領域では、逆の関係にあるので、分散記録しても影響は少ない。同じく、ベロシティ部分に記録する場合でも、ベロシティの値の小さい領域で、そのベロシティ部分に分散記録すると、それによる変動の割合（厳

密性）が大きくなるので、好ましくないが、ベロシティの値の大きい領域では、分散記録しても影響は少ない。従って、これらを適宜考慮して所定値よりも大きいベロシティに種々のデータを分散記録するようにしてもよい。

【0069】また、上述の実施の形態では、MIDIデータに分散記録する場合について説明したが、これ以外の波形データやシーケンスデータ、デジタル記録音声データ、画像データ、動画データ、レジストレーション（電子楽器の設定記録）データなどに分散記録するようにしてもよい。波形データに分散記録する場合には、所定の関数処理としてその波形データ（wave_data）そのもののモジュロ2、すなわち、 $f1(wave_data) = wave_data \bmod 2$ としてもよい。また、波形データの値とそのポイントデータとを用いて、両データの和のモジュロ2、すなわち、 $f2(wave_data, sample_point) = (wave_data + sample_point) \bmod 2$ としてもよい。図10及び図11の実施の形態では、各（論理）チャンネル毎に関数処理の種類を変更して、署名データを検出する場合について説明したが、プログラムチェンジデータの検出タイミング毎に関数処理の種類を変更するようにしてもよい。この場合には、検出されたデータ列に有意な部分が存在するかどうかの判定処理も各プログラムチェンジデータの検出タイミング毎に行う必要がある。

【0070】上述の実施の形態では、検出された電子署名データをモニタ上に表示する場合について説明したが、検出側において、電子署名データが存在しない場合や電子署名データが完全に復旧できない場合（各論理チャンネルにおいて、電子署名データが検出できるチャンネルと検出できないチャンネルが混在する場合）には、データの再生を中止するようにしてもよい。また、その検出側の電子楽器やコンピュータなどがネットワークに接続されている場合には、そのネットワーク上に不正にエディットされたデータが存在することを送信し、それをホストコンピュータ側で検出することができるようにしてもよい。上述の実施の形態では、電子楽器がMIDIデータ編集装置又はMIDIデータ修復装置として動作する場合について説明したが、同様の処理を行うハードウェアを別途構成するようにしてもよいし、ソフトウェア、DSPとマイクロプログラムなどで構成するようにしてもよい。また、このようなMIDIデータ編集装置又はMIDIデータ修復装置をフロッピーディスクドライブや通信インターフェイスなどに予め内蔵しておいて、データの入出力の段階で強制的に電子署名データを付与したり、検出したりするようにしてもよい。

【0071】上述の実施の形態では、電子署名データやスクランブルデコードデータを付与する場合を説明したが、電子署名データやスクランブルデコードデータの付

与されたデータを、フロッピーディスクやコンパクトディスクなどで供給するようにしてもよいし、ネットワークを通じて電子的に供給するようなデータ形態を採用してもよい。図8のステップ8Bでは、ベロシティデータの値を1だけインクリメント処理する場合について説明したが、デクリメント処理してもよいし、これらの処理を適当なタイミング(所定データ数毎とか)で交互に行うようにしてもよい。上述の実施の形態では、所定の関数処理の後にビット反転処理を行う場合について説明したが、これは行わなくてもよい。また、ビット反転処理の他にも、上位ビットと下位ビットを入れ替えるとか、AND、OR、XORなどの論理演算処理を施すとか、種々の処理を加えるようにしてもよいことはいうまでもない。また、適宜の暗号化処理を施すようにしてもよい。例えば、上述の第1から第4までの関数処理の結果を適宜組み合わせてもよいし、前回の関数処理の結果を次の関数処理に組み合わせる演算するようにしてもよい。なお、これらの各処理はある程度高速に行うことができ、変換後、逆変換によりもとに戻せるものであれば、これ以外の方法でもよいことはいうまでもない。

【0072】上述の実施の形態では、プログラムチェンジデータの検出タイミング毎に新たに電子署名データやスクランブルデコードデータの付与処理を行う場合について説明したが、これに限らず、小節単位(例えば8小節毎など)データの検出タイミング毎に行うようにしてもよい。また、波形データに分散記録する場合には、所定のサンプル数毎や所定のゼロクロス毎に新たに署名データの付与処理を行うようにしてもよい。ゼロクロス(波形の値の符号が変わるタイミング)は波形エディットの際の基準地点と考えられるからである。また、上述の実施の形態では、電子署名データや題名などのような付属情報を記録する場合について説明したが、MIDIデータと波形データを合わせて記録する場合に、MIDIデータにその波形データの一部を記録するようにしてもよいし、波形データにMIDIデータの一部を合わせて記録するようにしてもよい。

【0073】なお、上述の実施の形態では、署名データをMIDIデータのベロシティ部分に分散記録する場合について説明したが、これは一例であり、これ以外の著作者名、曲の題名(曲名)、画像/映像の題名などに関するテキスト情報や、そのデータ形式に関する情報やその他の種々の電子情報(曲の解説、著作者に関する情報、著作年、歌詞、ニュース、ハードID(機種名、使用OS)などの各種情報)を分散記録してもよいことはいうまでもない。この場合には、図8から図11までの各処理における電子署名をこれらの各種電子情報に置き換えて処理すればよい。このようにして処理された電子情報をモニタ画面上に表示するようにしてもよい。なお、このとき、付属情報を一連の纏まったファイル情報として検出して、これをメモリ等に記憶するあるいは適

宜これを利用する、ようにしてもよいし、自動演奏中にMIDIデータから検出される電子情報の流れとして認識し、それをリアルタイムに画面表示したりしてもよい。この場合、付属情報が画像情報の場合には、主要情報であるMIDIデータの受信に伴ってリアルタイムに画像が描画されるようになり、テキスト情報の場合には、ニュース配信などのようなリアルタイムな情報表示を行うことが可能となる。また、情報を付加するための技術がシンプルであるため、オンデマンドなどで電子情報を埋め込むことができるので、TV放送などのようなリアルタイムな描画が可能となる。また、通信ネットワークを通じた情報通信において本発明を適用してもよく、例えば、インターネットや電子メールなどのIDやパスワードの配信にも利用することができる。また、簡単なサウンドメッセージを付属情報としてもよい。

【0074】上記のような、通信ネットワークを通じた本発明の実施の形態を略示するシステム構成図が、図16に示されている。送信側において、送信しようとする主要情報(例えばMIDI情報のような音楽演奏情報)がサーバー160から供給される。スクランブルエンコード161では、本発明の各実施例に示したやり方で、この主要情報に対して任意の付属情報がリアルタイムで組み込まれる。付属情報を適宜組み込んだ主要情報がネットワーク163を介して送信される。受信側では、デコード164において、本発明の各実施例に示したやり方で、受信したデータから付属情報が組み込まれている部分を検出し、これをデコードして、付属情報と主要情報を分離して再生する。MIDI情報のような音楽演奏情報からなる主要情報は音源165に与えられて、音楽演奏を再生する。テキストや静止画像あるいは動画等からなる付属情報はディスプレイ166に与えられて、リアルタイムで表示される。スクランブルエンコード161及びデコード164は、上述のようにソフトウェアで構成されていてもよいし、あるいは専用ハードウェアで構成してもよい。

【0075】なお、上述の第1の実施の形態、すなわち図3のMIDIデータ編集処理1では、ステップ38において、前回のキーオンイベントデータのキーコードを格納した前回値レジスタb0と今回のキーオンイベントデータのキーコードを格納したキーコードレジスタbとの間の差分を取り、それをキーコード差分値レジスタdkeyに格納すると共に、前回のキーオンイベントデータのベロシティを格納した前回値レジスタc0と今回のキーオンイベントデータのベロシティを格納したベロシティレジスタcとの間の差分を取り、それをベロシティ差分値レジスタdve1に格納している。その関係上、ステップ3Aからステップ3Gまでの処理によって変換された新しいキーコードの値b1及びベロシティの値c1は、常に前回のキーオンイベントにおけるキーコード及びベロシティとの間に密接な関係を有することにな

る。図4のMIDIデータ修復処理1でも、前回値レジスタb0とキーコードレジスタbとに基づいて修復キーコードレジスタb2の値を求め、前回値レジスタc0とベロシティレジスタcとに基づいて修復ベロシティレジスタc2の値を求めている。従って、図3のMIDIデータ編集処理後にMIDIデータのイベントデータ（キーコード又はベロシティの値）が修正されると、それ以降に修復処理された修復キーコード及び修復ベロシティの値が元のデータと全て異なってしまうということが起こる。これはデータを改竄した場合にそれ以降のデータを全て無効にすることができるという点では優れているが、途中のイベントデータにエラーが発生し、キーコード又はベロシティが変化した場合、それ以降のデータを有効に修復することができなくなるという問題を有する。

【0076】そこで、図3のMIDIデータ編集処理1及び図4のMIDIデータ修復処理1を図13のMIDIデータ編集処理4及び図14のMIDIデータ修復処理4のように変更することによって、イベントデータが途中で書き換えられたりなどして変化した場合でも、それ以降のデータを有効に修復することができるようにした。図13のMIDIデータ編集処理4において図3のMIDIデータ編集処理1と同じものには同一の符号が付してあるので、その説明は省略する。図13のMIDIデータ編集処理4が図3のMIDIデータ編集処理1と異なる点は、図3のステップ39の処理が省略され、ステップ38の前回値レジスタb0及びc0がステップ138のように先頭値レジスタbH及びcHに変更された点である。ステップ138では、先頭のキーオンイベントデータのキーコードを格納した先頭値レジスタbHと今回のキーオンイベントデータのキーコードを格納したキーコードレジスタbとの間の差分を取り、それをキーコード差分値レジスタdkeyに格納すると共に、先頭のキーオンイベントデータのベロシティを格納した先頭値レジスタcHと今回のキーオンイベントデータのベロシティを格納したベロシティレジスタcとの間の差分を取り、それをベロシティ差分値レジスタdvelに格納する。そして、このキーコード差分値レジスタdkeyの格納値とベロシティ差分値レジスタdvelの格納値に基づいて新たなキーコードb1及びベロシティc1を算出している。

【0077】図14のMIDIデータ修復処理4において図4のMIDIデータ修復処理1と同じものには同一の符号が付してあるので、その説明は省略する。図14のMIDIデータ修復処理4が図4のMIDIデータ修復処理1と異なる点は、図4のステップ48の処理が省略され、ステップ45及びステップ46の前回値レジスタb0及びc0がステップ145及びステップ146のように先頭値レジスタbH及びcHに変更された点である。すなわち、ステップ145では先頭値レジスタbH

とキーコードレジスタbとの値に基づいて修復キーコードレジスタb2の値を求め、ステップ146では先頭値レジスタcHとベロシティレジスタcとの値に基づいて修復ベロシティレジスタc2の値を求めている。図13及び図14のように変更することによって、先頭のイベントデータの値に基づいてMIDIデータは編集され、修復されるようになるので、途中のイベントデータが書き換えられたり、エラーなどの発生によって変化した場合でもそれ以降のデータを修復できなくなるということとは起こらず、エラーの発生したイベントデータ以降も完全に修復することができるようになる。なお、上述の説明では先頭のイベントデータの値に基づいてMIDIデータを編集する場合について説明したが、予め任意の値をヘッダに書き込んでおいて、その値に基づいてMIDIデータを編集したり、修復したりするようにしてもよい。また、その基準となるイベントデータの値を任意のイベント（例えば1小節における先頭のイベント）から取り込むようにしてもよい。

【0078】ところで、図14のMIDIデータ修復処理において、レジスタbH、cHにストアされるデータは、電子透しが埋め込まれた又は暗号化されたデータを修復するための「鍵情報」の一種に相当している。このデータ修復のための「鍵情報」は、上記実施例では先頭のキーオンイベントデータとして主要情報（電子透しまたは暗号化の対象となる情報）の中に含まれているものが用いられているが、これに限らず、データ修復のための「鍵情報」が、主要情報にも付属情報にも含まれないようにしてもよい。すなわち、図14のMIDIデータ修復処理において、レジスタbH、cHにストアされるデータ修復用「鍵情報」を、主要情報及び付属情報のデータファイルとは別途に供給するようにする。勿論、その場合、レジスタbH、cHにストアされるデータ修復用「鍵情報」は、上記のような先頭のキーオンイベントデータのキーコード又はベロシティデータである必要はなく、適宜に設定したデータであってよい。そのためには、データ編集処理（つまり電子透し埋め込み又は暗号化すなわちエンコード処理）の際に、例えば図13のステップ138で使用するレジスタbH、cHに格納するデータとして、所望のデータを任意に設定するようにすればよい。このレジスタbH、cHに格納するデータは前述の通り、電子透し埋め込み又は暗号化（つまりエンコード）のための所定のアルゴリズムにおいて変数の一種（エンコードしようとするデータの差分値を算出するための演算の変数）として、すなわち電子透し埋め込み又は暗号化演算のための補助情報として、使用されるものである。従って、所望の変数を設定入力し、これをレジスタbH、cHにストアすることで、該設定入力した変数を用いて電子透し埋め込み又は暗号化（つまりエンコード）処理が行われるようにすればよい。この変数（つまり、データ修復の際には「鍵情報」として利用さ

れるもの)の設定入力処理は、図13のステップ138のあたりで行うようにしてもよいが、その前の段階、例えば、ステップ31のあたりで、行うようにするとよい。勿論、この変数(つまり、データ修復の際には「鍵情報」として利用されるもの)の設定入力処理は、手動入力に限らず、データ通信その他適宜の手段を介して自動で行うようになっていてもよい。

【0079】このようにすることにより、データ修復時においてレジスタbH, cHにストアすべきデータ修復用「鍵情報」を具備していない(若しくは知らない)者が、不当にデータを修復しようとする場合、その修復アルゴリズムが判明していたとしても、肝心のデータ修復用「鍵情報」を得ることができないことになり、データ修復を行なうことができないことになり、暗号化又は電子透しによる機密情報埋め込みに際して、セキュリティ機能を向上させることができる。なお、この場合、レジスタbH, cHにストアされるデータ修復用「鍵情報」を主要情報及び付属情報のデータとは別途に供給する際の、該データ修復用「鍵情報」の供給の仕方は、どのようなやり方を用いてもよい。例えば、別途のデータ通信によって行なうようにしてもよいし、印刷物で通知してもよいし、電話等で通知してもよいし、様々なやり方であってよい。正当な利用者に対して秘密裡に通知又は供給されたデータ修復用「鍵情報」は、自動的に又は利用者による手動入力によってレジスタbH, cHにセットされるようになっていてよく、こうして、正当な利用者だけが図14のようなデータ修復処理を正当に遂行し、データ修復を成功裡に行なうことができる。

【0080】そのために、例えば、図14に示すような「データ修復処理4」の手順を図17に示す「データ修復処理4'」のように変更するとよい。図17は、ステップ41の前にステップ40が挿入されている点だけが図14とは異なっており、他は同一構成であってよい。このステップ40において、上記のように適宜入力された所定の鍵情報を取り込み、それに基づき必要なデータを各レジスタbH, cHにロードする。なお、この実施例の変形例として、レジスタbH, cHにストアされるべきデータ修復用「鍵情報」の一部を、主要情報及び付属情報のデータファイルとは別途に供給するようにし、残りは主要情報及び付属情報のデータファイルの中に適宜含ませるようにしてもよい。その場合は、別途に供給された「鍵情報」の一部とデータファイルの中に含まれた残りの部分とを合成することで、レジスタbH, cHにストアされるべきデータ修復用「鍵情報」を生成する。その場合、鍵情報を合成するステップは少なくともステップ45の手前に設定するものとし、合成したデータを各レジスタbH, cHにロードするものとする。

【0081】ところで、図3のMIDIデータ編集処理1及び図13のMIDIデータ編集処理4では、キーオンデータのキーコード及びベロシティの内容がMIDI

データの先頭から書き換えられてしまうので、MIDIデータ修復処理を行わずにMIDIデータ再生処理を行った場合に、再生された楽音は、全くデタラメなものとなってしまう、その一部だけを試聴するというようなことができない。そこで、図3の変更例として、図15のMIDIデータ編集処理5のように、ステップ3Bとステップ3Cとの間に編集条件を判定するステップ3Jを挿入し、特定の編集条件に合致した場合にMIDIデータにスクランブル処理を施すようにするとよい。すなわち、電子透かし埋め込み又は暗号化(すなわちエンコード)処理のためのアルゴリズムにおいて使用される変数の一種として、すなわち電子透し埋め込み又は暗号化演算のための補助情報として、「編集条件」設定情報が追加されることになる。なお、図3の変更例である図13の場合も、図15と同様に、ステップ3Bとステップ3Cとの間にステップ3Jを挿入するように変形し、電子透かし埋め込み又は暗号化(すなわちエンコード)処理のためのアルゴリズムにおいて使用される変数の一種として「編集条件」の変数を追加するようにしてよい。

【0082】例えば、ステップ3Jの編集条件として、MIDIデータの演奏開始から所定時間(例えば30秒などの時間)を経過したか否かを判定するように設定する。これによって、MIDIデータの最初の部分にはスクランブル処理が施されなくなるので、その部分のMIDIデータは通常のMIDIデータ再生処理によって再生されるようになる。一方、所定時間経過後にMIDIデータを通常のMIDIデータ再生処理で再生しようとしても、その部分にはスクランブル処理が施されているので、デタラメな楽音しか再生されなくなる。これによって、MIDIデータの最初の部分を通常のMIDIデータ再生処理によって再生できる試聴可能なMIDIデータとすることができ、それ以降の残りの部分を試聴不可能なMIDIデータとすることができる。なお、この「編集条件」変数は、外部から任意に設定可能であり、時間の他にも、種々の条件を適宜組み合わせて適用してもよいことは言うまでもない。例えば、発音されない時間が所定時間以上になった場合とか、小節線データが所定数発生した場合とかの条件を単独で適用したり、これらの条件を種々組み合わせてもよいことは言うまでもない。

【0083】また、例えば、MIDIデータファイルの先頭から電子透し埋め込み又は暗号化のための編集操作を開始するまでのデータ数及びそこから実際に編集を行なったデータ数を「編集条件」変数として設定する(すなわちMIDIデータファイルのどの箇所に電子透し埋め込み又は暗号化のためのデータ編集操作を施したかを具体的に編集条件として設定する)ようにしてもよく、そのようにすれば、MIDIデータファイルの任意の箇所に対して電子透し埋め込み又は暗号化のためのデータ編集操作を加えることができる。なお、所望の編集条件

の設定処理は、ステップ3 Jで行ってもよいが、その前の段階、例えばステップ3 Iのあたり、で所望の編集条件を設定入力し、これを取り込むようにすればよい。また、設定する編集条件は一種類に限らず、複数種類であってもよい。

【0084】なお、データ修復処理の際には、暗号化（つまりエンコード）の際に用いられた編集条件がどのようなものであるかを示す情報を「鍵情報」として使用することで、正しい編集条件「鍵情報」を持つものだけがデータ修復を行えるようにするとよい。例えば、これらの編集条件をユーザー毎にあるいはMIDIデータファイル毎に異ならせて設定してデータ編集処理（つまり電子透かし埋め込み又は暗号化のためのエンコード処理）を行った上で、それぞれの編集条件を示す情報を、個別のデータ修復用「鍵情報」として別途適宜に供給するようにし、正当なる利用者がデータ修復時に該データ修復用「鍵情報」を正当に取得して、その「鍵情報」が示す編集条件と修復すべきデータにおける実際の編集条件とが合致したときにデータ修復処理が遂行されるようにするとよい。このようにすることにより、データ修復のアルゴリズムは共通であっても、この「鍵情報」の多様化によって、電子透かし埋め込み又は暗号化の形態を多様化させることができ、信頼性／セキュリティの向上を図ることができる。

【0085】なお、この場合のデータ修復用「鍵情報」を別途に供給するやり方は、前述の場合と同様に、どのようなやり方を用いてもよい。また、データ修復処理に際しての編集条件の照合の仕方としては、例えば、データ修復しようとする利用者が所要の「鍵情報」を入力することによって編集条件を設定入力し、この設定入力された編集条件に該当するデータファイル箇所において図15のステップ3 C～3 Fに示すデータ暗号化処理の逆算であるデータ修復（デコード）処理を行なうようにする。ここで、設定入力された編集条件が正しければ該データ修復処理では正しくデータ修復がなされることになり、正しくなければ該データ修復処理では正しいデータ修復が行なえないことになる。なお、この場合の「鍵情報」の入力の仕方も、前述と同様に手動入力に限らず自動的になされるようにしてもよい。図18は、図4に示すような「データ修復処理1」を、編集条件を鍵情報として使用してデータ修復を行うのに適したものに变形した一例（「データ修復処理5」）を示す図である。図18において、ステップ41の手前にステップ40'を挿入し、ステップ41と42の間にステップ410を挿入した点が図4とは相違しており、他は図4と同じである。ステップ40'では、編集条件の「鍵情報」の入力を受け付け、これを取り込む。ステップ410では、取り込んだ「鍵情報」によって示される編集条件が成立したか否かを調べる。編集条件が成立していなければ、ステップ42以降のデータ修復処理は行わずにステップ4

Cに行き、そのNOからステップ41に戻る。取り込んだ「鍵情報」によって示される編集条件が成立した場合は、ステップ42～4Bで所要のデータ修復処理を行う。なお、他の「データ修復処理」（例えば図14）に対しても、図18と同様の变形を施すことができる。

【0086】なお、上記のようにデータ修復用「鍵情報」は一種類とは限らず、複数種類あってもよいものである。また、同じ種類の鍵情報でも異なる値のものが複数あってもよい。その場合、例えば鍵管理ファイルをデータファイルとは別途に用意しておき、この鍵管理ファイルから必要な「鍵情報」を正当な利用者だけが取得できるように管理するとよい。例えば、利用者は所定のアルゴリズムでデータファイルから鍵取得情報を抽出し、その鍵取得情報を入力情報として鍵管理ファイルから必要な鍵情報を取得できるようにする。この場合、鍵管理ファイルは入力された鍵取得情報に応じて適宜異なる鍵情報を供給するものとする。このようにして複数種類の必要な鍵情報をそれぞれ取得できるようにし、これらの鍵情報を使用して上記のようにデータ修復を行うようにする。ここで、更にデータファイル内の複数の箇所複数異なる鍵情報を取得しなければ適正なデータ修復が行なえないようにすれば、より厳密な管理とセキュリティ保持を行なうことができる。なお、鍵管理ファイルの部分は通信ネットワークのサーバー上に格納しておき、データ通信によってオーソライズされた利用者だけが鍵取得情報を入力して、これに応じてサーバーから必要な鍵情報をデータ通信によって随時取得することができるようにしてもよい。

【0087】

【発明の効果】この発明によれば、音楽データ、映像データ又は波形データなど主要情報のデータフォーマットを変更することなく所望の付属情報を潜在的に付加してなるデータファイルを提供することができると共に結果的にこれらのデータに暗号化処理を施すこととなり、この暗号を解読しない限り、これらの主要情報や付属情報を再生して利用することができないという効果がある。また、主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更するために、主要情報における変更すべき前記一部のデータに対して、所定のアルゴリズムに従う演算を、前記付属情報を構成するデータと外部から設定可能な補助情報とを用いて行なうようにしたことで、解読の際の鍵情報となりうる補助情報そのものは、データファイルのデータ内容とは無関係に任意に設定されるから、該データファイルの中には直接的にはそれと判るようには含まれていないものとしてすることができるものであり、これにより、オーソライズされていない者が、不当に当該データファイルから主要情報の修復を行ないまたその付属情報を解読しようとしたとしても、肝心のデータ修復用の「鍵情報」をデータファイルから得ることができないことによ

り、データ修復若しくは解読を行なうことができないことになり、暗号化又は電子透し埋め込みによる機密的な情報記録に際して、セキュリティ面での信頼性を向上させることができる、という優れた効果を奏する。

【図面の簡単な説明】

【図1】 この発明に係る電子情報処理システムのMIDIデータ編集処理1及びMIDIデータ修復処理1に従ってどのようにデータが変換されるのか、その具体例を示す図である。

【図2】 この発明に係る電子情報処理システムとして動作する電子楽器の全体構成を示すブロック図である。

【図3】 図2の電子楽器が電子情報処理システムとして動作する場合のMIDIデータ編集処理1の一例を示すフローチャート図である。

【図4】 図2の電子楽器が電子情報処理システムとして動作する場合のMIDIデータ修復処理1の一例を示すフローチャート図である。

【図5】 縦軸にキーコードを割り当て、横軸にベロシティを割り当て、各値の中間を原点とするような直交座標系を構成した場合の付属情報とキーコード、ベロシティとの関係を示す概念図である。

【図6】 図5の付属情報とキーコード、ベロシティとの間の詳細な関係を示す図である。

【図7】 MIDIデータ中に埋め込まれるスクランブルデコードデータの内容の一例を示す図である。

【図8】 電子楽器が電子情報処理システムとして動作する場合の別の一例となるMIDIデータ編集処理2を示すフローチャート図である。

【図9】 電子楽器が電子情報処理システムとして動作する場合の別の一例となるMIDIデータ修復処理2を示すフローチャート図である。

【図10】 電子楽器が電子情報処理システムとして動作する場合の別の一例となるMIDIデータ編集処理3を示すフローチャート図である。

【図11】 電子楽器が電子情報処理システムとして動

作する場合の別の一例となるMIDIデータ修復処理3を示すフローチャート図である。

【図12】 この発明に係る電子情報処理システムのMIDIデータ編集処理2及びMIDIデータ修復処理2に従ってどのようにデータが変換されるのか、その具体例を示す図である。

【図13】 電子楽器が電子情報処理システムとして動作する場合の別の一例となるMIDIデータ編集処理4を示すフローチャート図である。

【図14】 電子楽器が電子情報処理システムとして動作する場合の別の一例となるMIDIデータ修復処理4を示すフローチャート図である。

【図15】 電子楽器が電子情報処理システムとして動作する場合の別の一例となるMIDIデータ編集処理5を示すフローチャート図である。

【図16】 通信ネットワークを通じて実施する本発明の一実施の形態を例示するブロック図である。

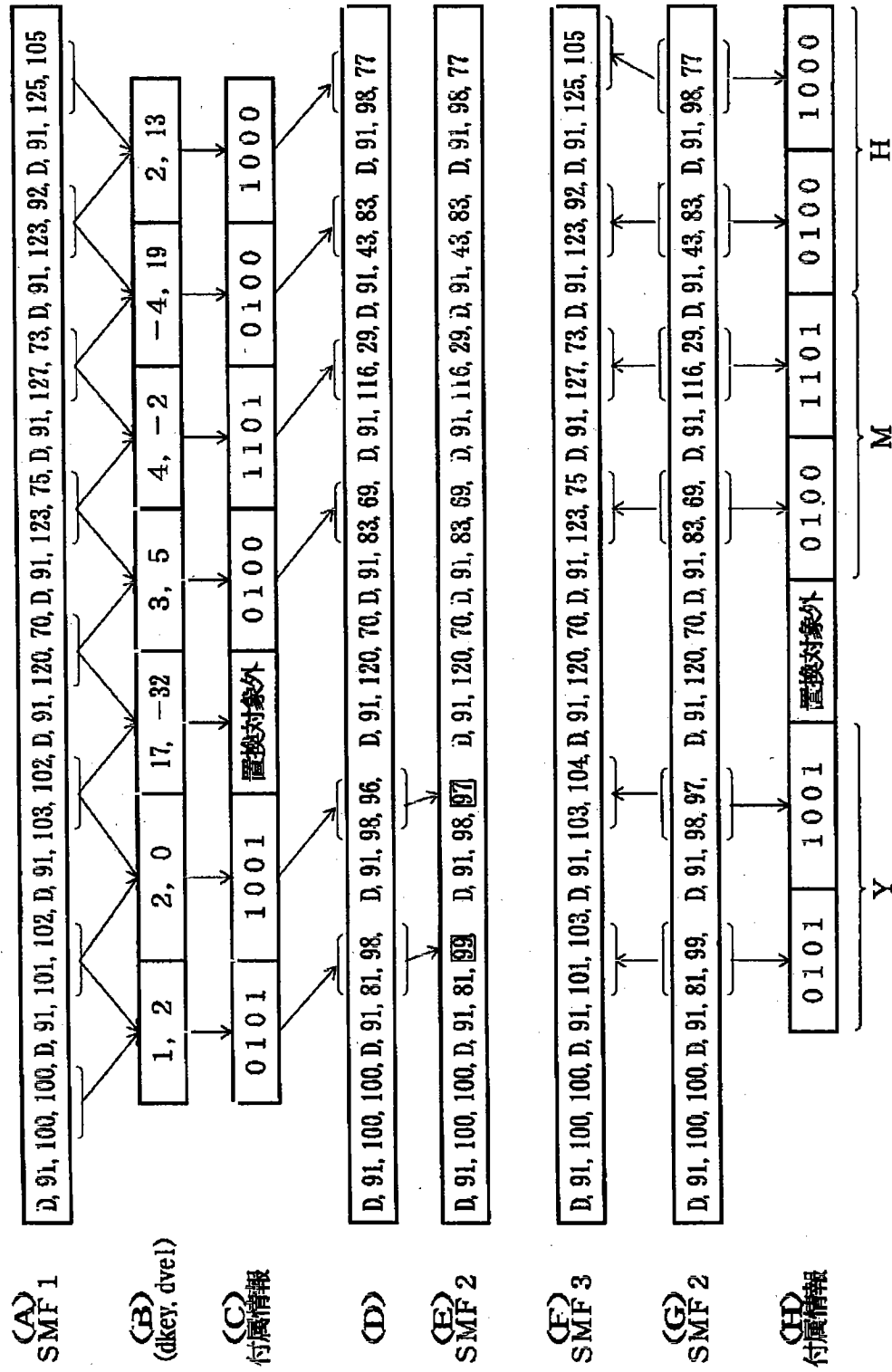
【図17】 図14のMIDIデータ修復処理の変形例を示すフローチャート図である。

【図18】 図15のMIDIデータ編集処理5で処理された情報のデータ修復に使用できるMIDIデータ修復処理5の一例を示すフローチャート図である。

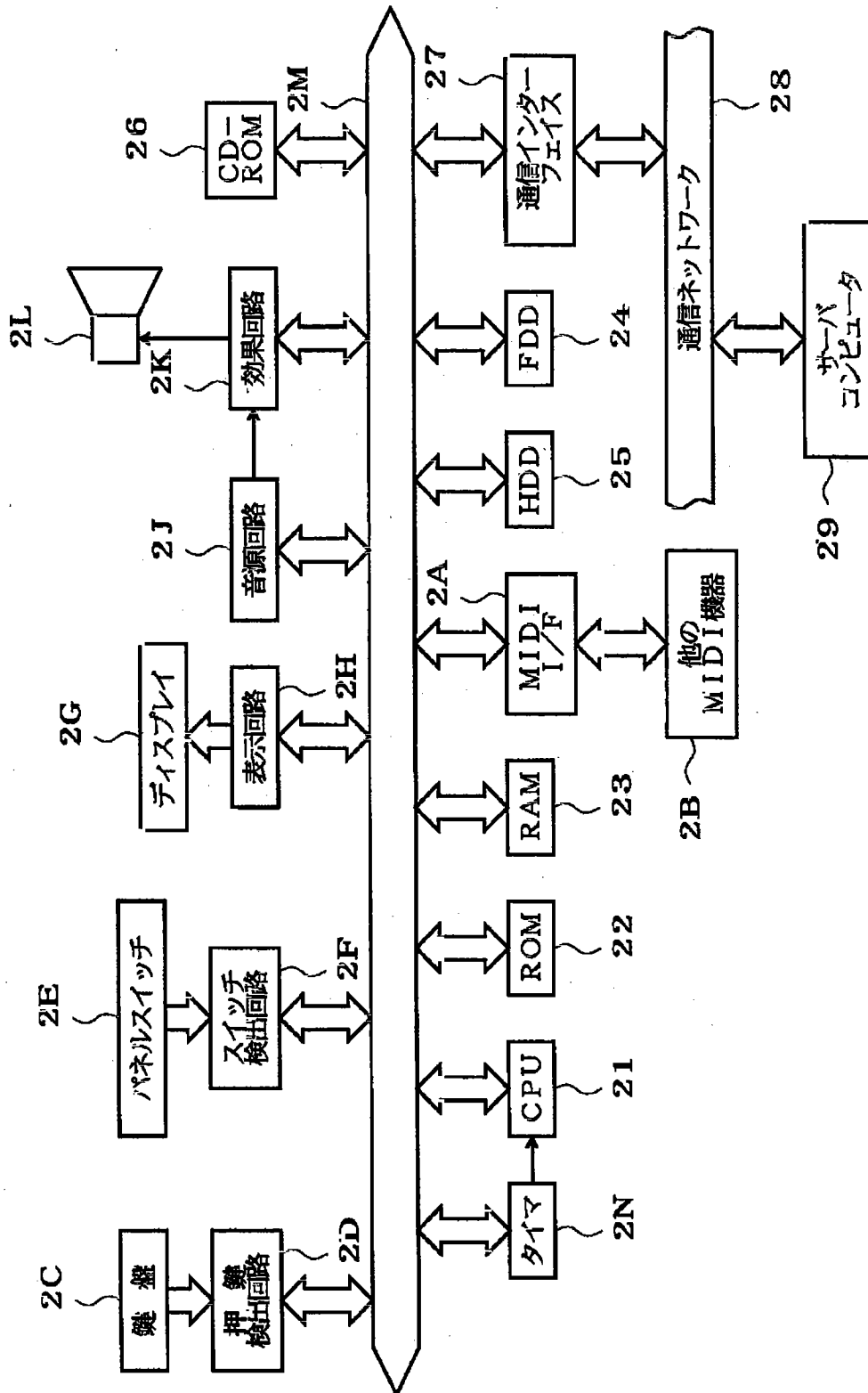
【符号の説明】

21…CPU、22…ROM、23…RAM、24…フロッピーディスクドライブ、25…ハードディスクドライブ、26…CD-ROMドライブ、27…通信インターフェイス、28…通信ネットワーク、29…サーバコンピュータ、2A…MIDIインターフェイス、2B…他のMIDI機器、2C…鍵盤、2D…押鍵検出回路、2E…パネルスイッチ、2F…スイッチ検出回路、2G…ディスプレイ、2H…表示回路、2J…音源回路、2K…効果回路、2L…サウンドシステム、2M…データ及びアドレスバス、2N…タイマ、2R…リボンコントローラ

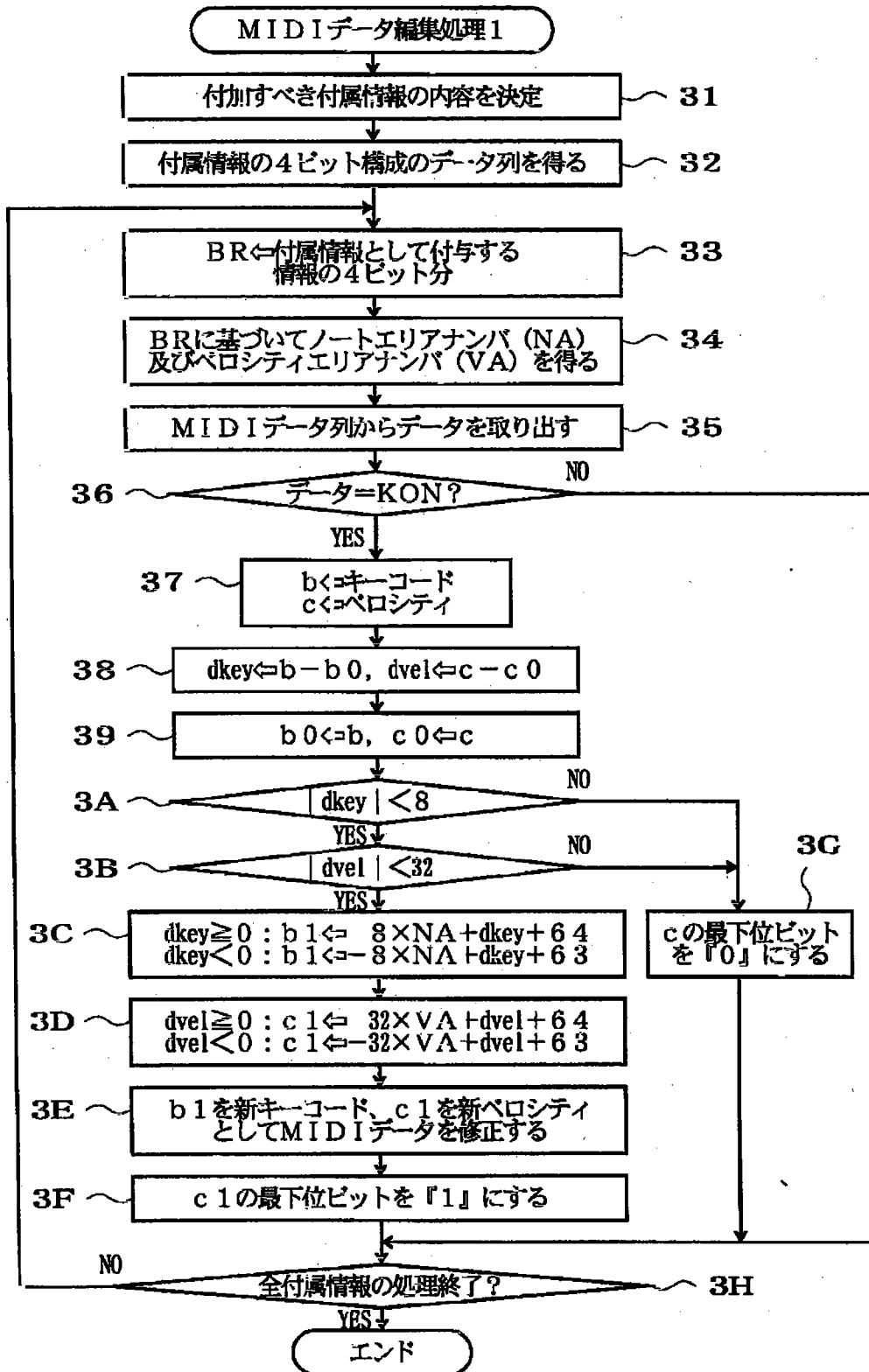
【図1】



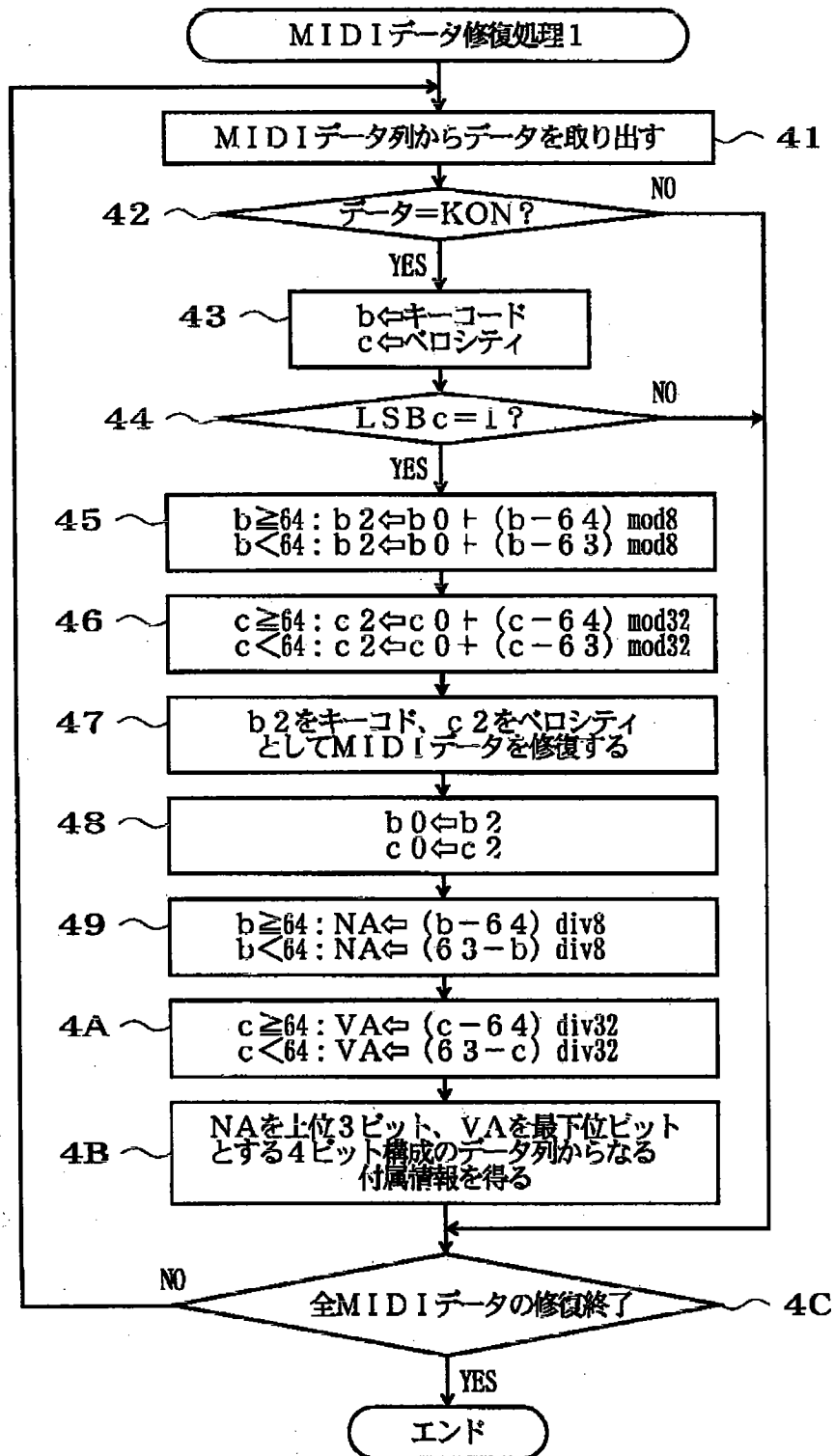
【図2】



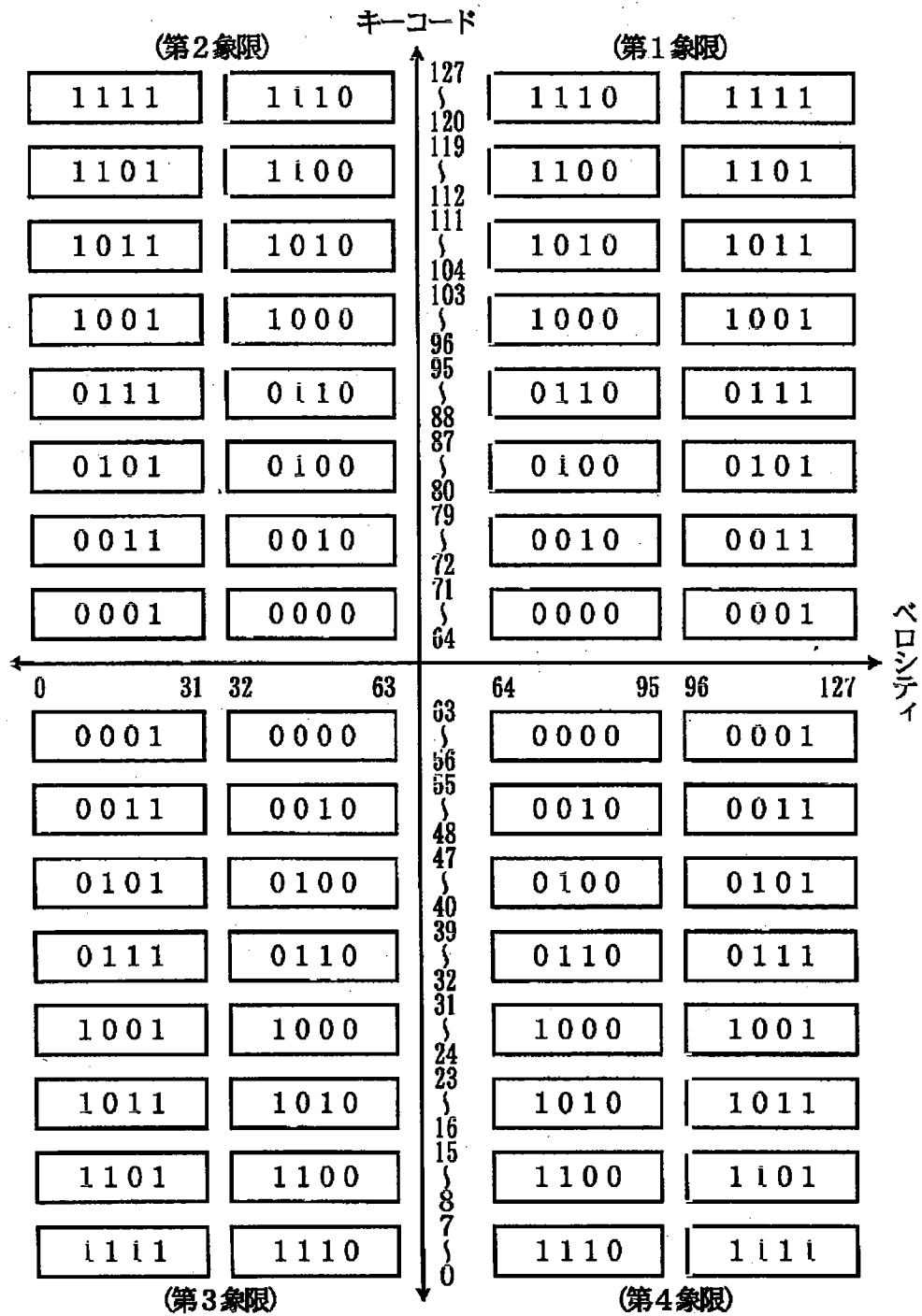
【図3】



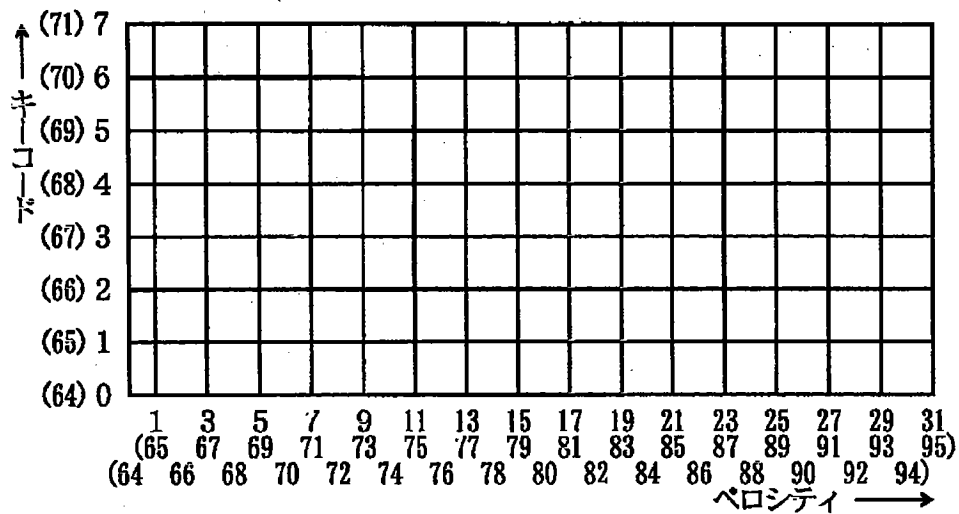
【図4】



【図5】



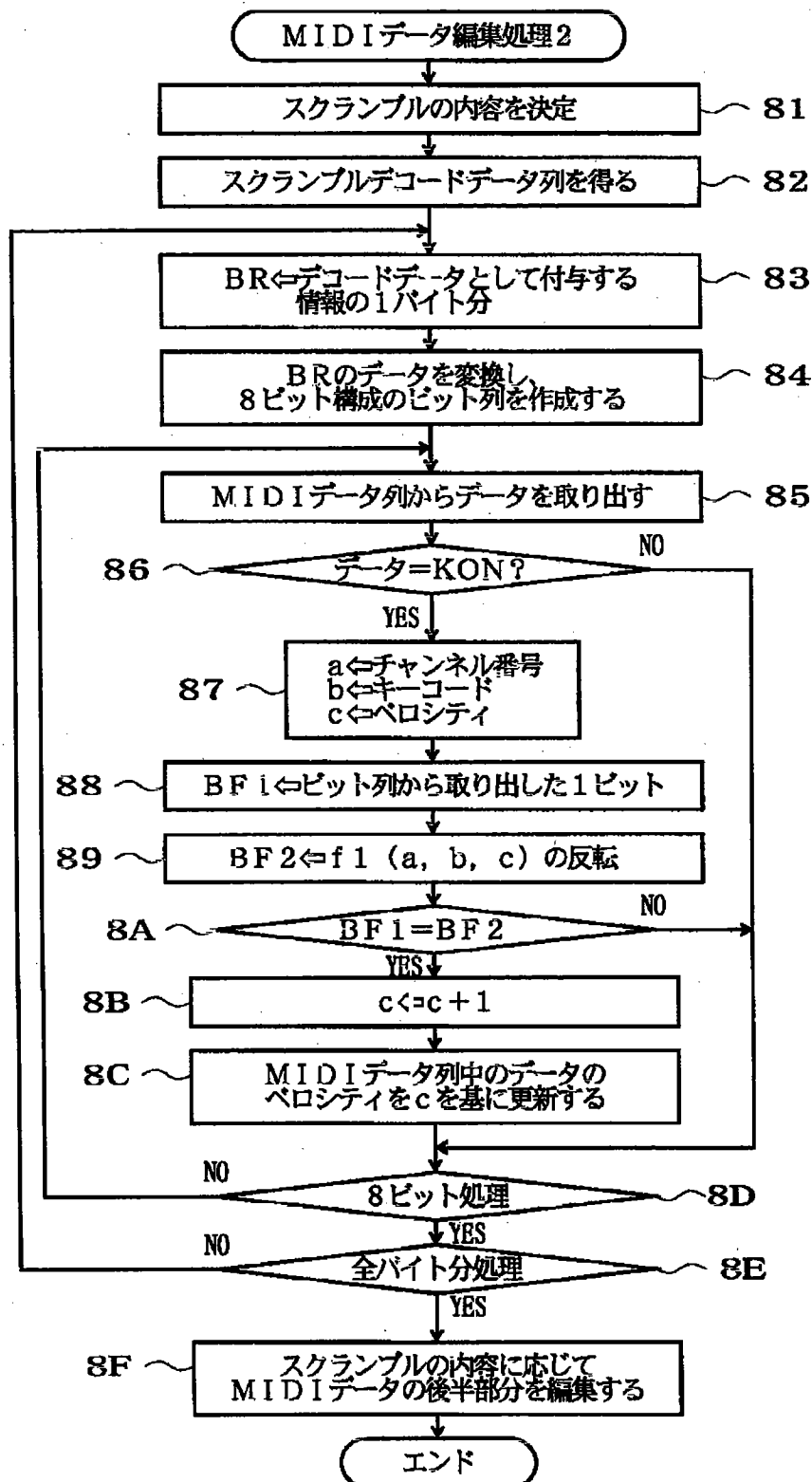
【図6】



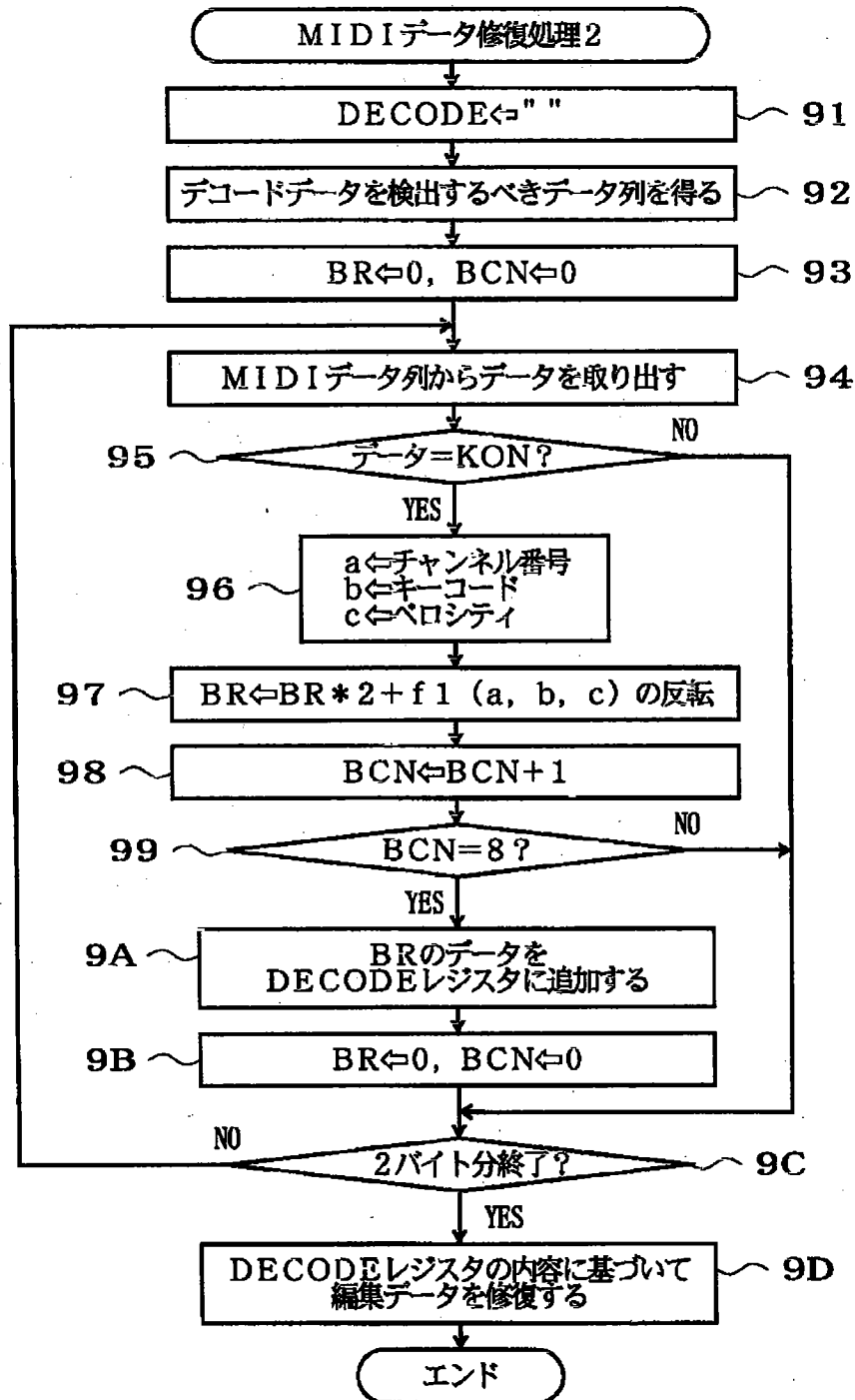
【図7】

4ビット	4ビット	8ビット
アルゴリズム	カウント数	Value
キーオンデータのノート番号を変更する	修正するイベント数	トランスポーズ値 (修正ノート値)
キーオンデータのノートとベロシティを入れ替える	修正するイベント数	なし
インターバルデータを変更する	修正するイベント数	修正値
キーオンデータのチャンネルデータを変更する	修正するイベント数	修正チャンネル
その他		

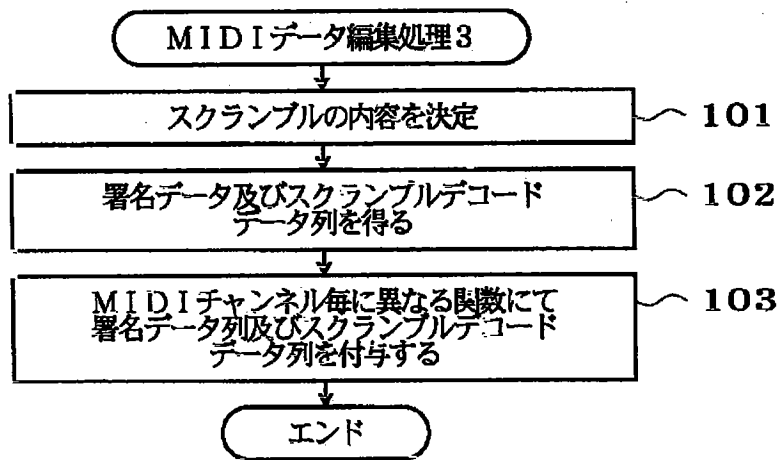
【図8】



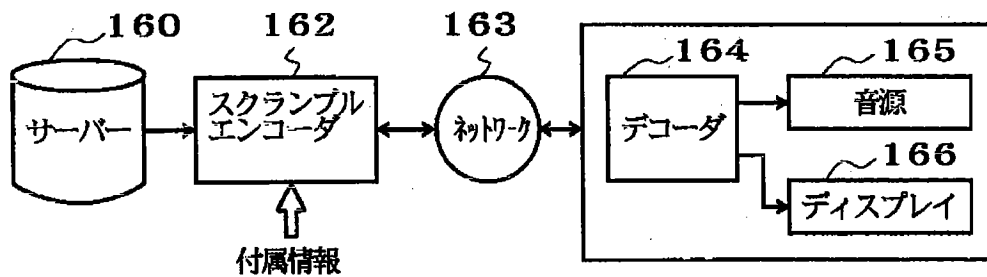
【図9】



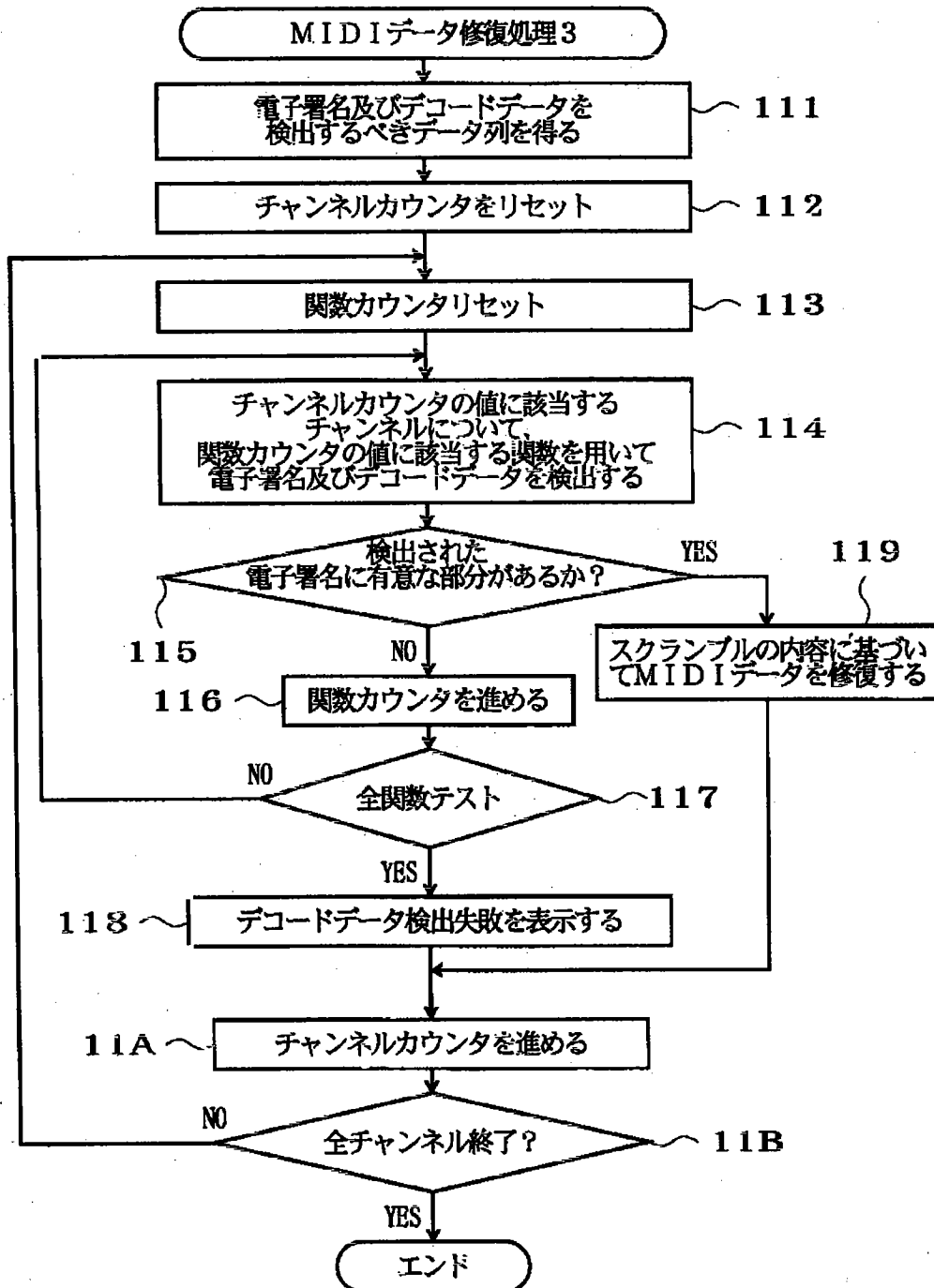
【図10】



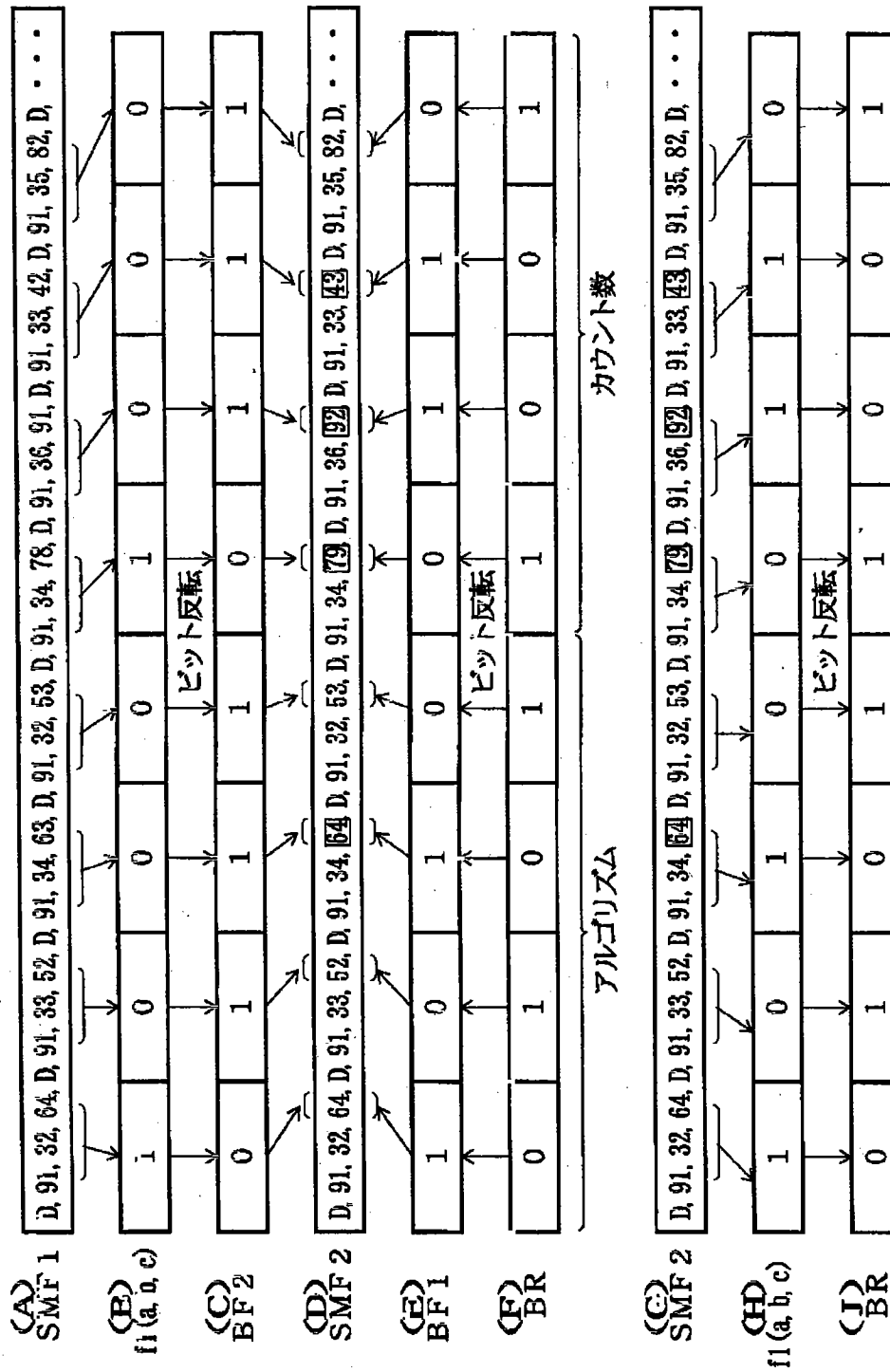
【図16】



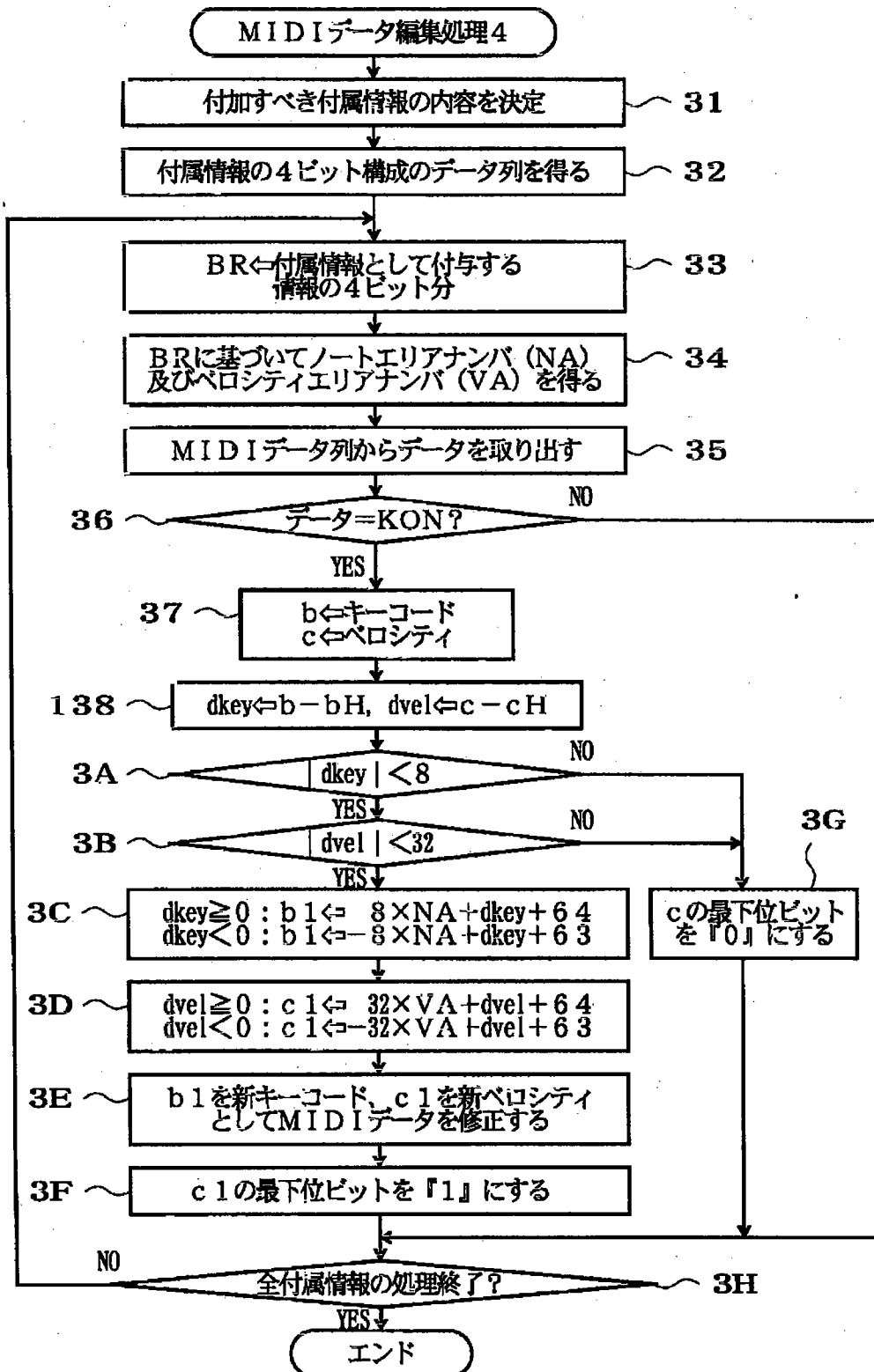
【図11】



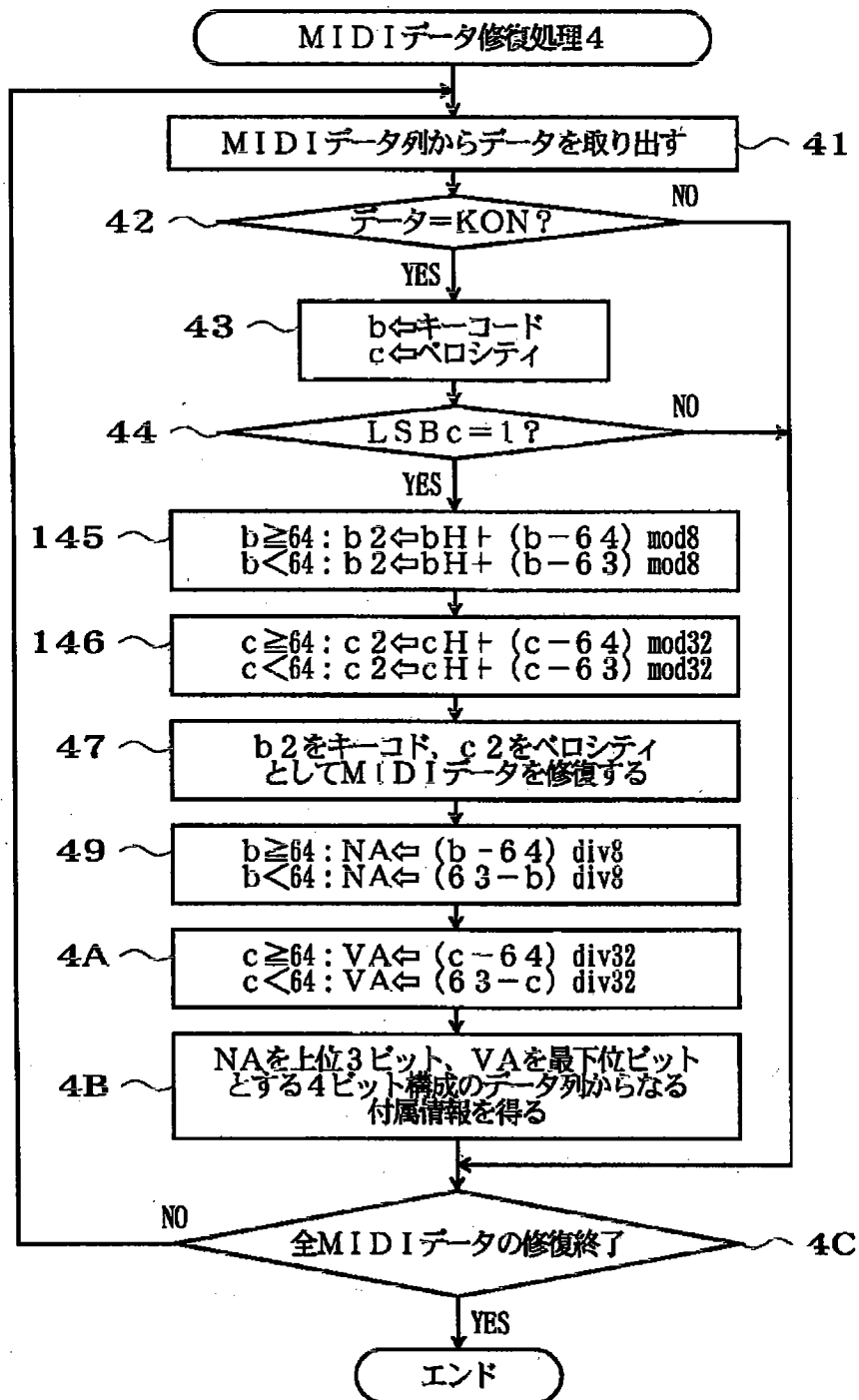
【図12】



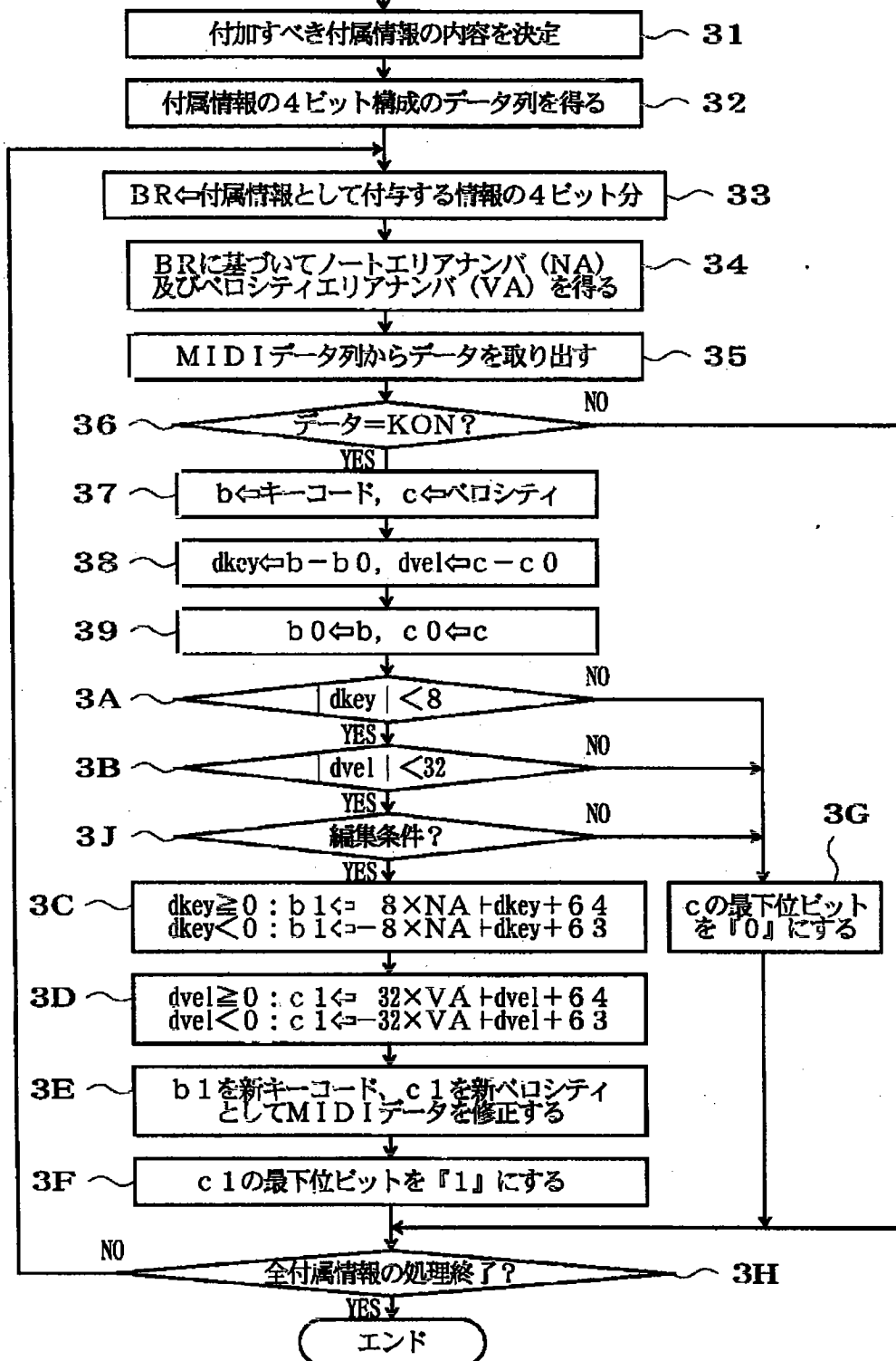
MIDIデータ編集処理4



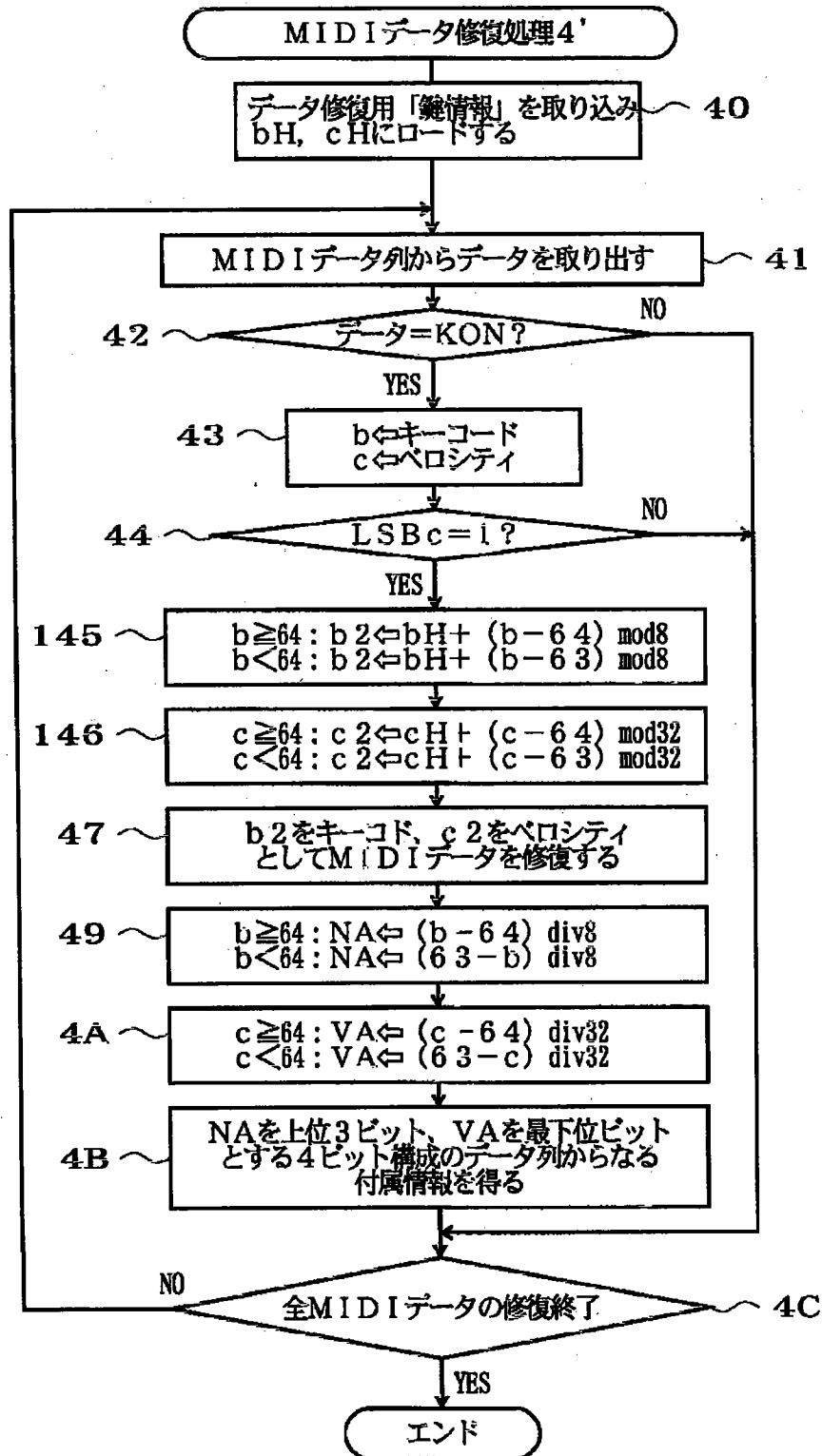
【図14】



MIDIデータ編集処理5



【図17】



【図18】

